J Intell Manuf (2013) 24:235–250
DOI 10.1007/s10845-011-0578-5

Survey on assembly sequencing: a combinatorial and geometrical perspective

P. Jiménez

Received: 26 November 2010 / Accepted: 16 July 2011 / Published online: 11 August 2011

Abstract A systematic overview on the subject of assembly sequencing is presented. Sequencing lies at the core of assembly planning, and variants include finding a feasible sequence—respecting the precedence constraints between the assembly operations—, or determining an optimal one according to one or several operational criteria. The different ways of representing the space of feasible assembly sequences are described, as well as the search and optimization algorithms that can be used. Geometry plays a fundamental role in devising the precedence constraints between assembly operations, and this is the subject of the second part of the survey, which treats also motion in contact in the context of the actual performance of assembly operations.

Keywords Assembly sequencing · Assembly optimization · Separability · Contact states

Introduction

An **assembly** $\mathcal{A}$ is an object composed of individual **parts** in given relative placements, such that they do not overlap and each part is touching a subset of the assembly. **Assembly sequencing** computes an ordering of collision-free operations that bring these parts together (assembly operations), given a geometric description of their positions in the final assembly product. Sequencing is the most important phase of the broader problem of **assembly planning**, which includes other topics like resource allocation, work-cell layout, or tolerance-related issues. Some works address specific instances of such extensions, like for example Wang et al. (2008) where assembly sequencing is treated concurrently with fixturing (i.e., holding the intermediate subassemblies in place).

Assembly sequencing is an obvious component of process planning. Nonetheless, its role in the early phase of product design is fundamental for optimizing not only the manufacturability of the product but also the design process itself. A feasible sequence validates a specific design, whereas optimality measures on different sequences allow to choose among various design alternatives.

An assembly can be considered at different levels of granularity, whose respective items are subassemblies, parts, features, and boundary primitives. Assembly sequencing is usually treated as a combinatorial problem, which deals with symbols or labels corresponding to elements from the two first levels. Not any combination of parts into subassemblies is allowed, **feasible** assemblies have to satisfy given **contact** (between parts) and **precedence** (between assembly operations) constraints. The so called *three step approach* of assembly sequencing comprehends the definition of precedence constraints, the generation of all feasible sequences and finally the choice among them. At the combinatorial level, precedence constraints are already given, or the means exist to provide them at specific request, but it is no matter of concern of how they are obtained. Actually, they constitute the output of a geometrical reasoning process that deals with boundary primitives or simple geometric features. This survey considers both the combinatorial and the geometrical aspects of an assembly. The first three sections are devoted to the components of assembly sequencing from the combinatorial point of view: representations of the space of possible sequences (section "Representing assembly sequences"), the criteria to be considered when selecting among various alternative sequences and the inherent complexity to different problem settings (section "On the complexity of sequencing"), as well as the sequencing and optimization algorithms

P. Jiménez (✉)
Institut de Robòtica i Informàtica Industrial (CSIC, UPC), Llorens i Artigas 4-6, 08028 Barcelona, Spain
e-mail: pjimenez@iri.upc.edu

(section "Sequencing and optimization algorithms"). As for the geometry of assembly sequencing, the second part of this survey reviews the analytical tools that allow to obtain the relative precedence constraints between parts from a geometrical description of the assembly. In most of this work, the reverse problem of disassembly is tackled. Section "Separability and blocking relationships between parts" uses the concept of blocking directions to determine subassemblies that can be separated from the whole assembly (or the remains, after other disassembly operations), section "From geometry to compliance: (dis)assembly as motion in contact" describes ways to represent the space of contact spaces and how to plan at high (symbolic) and low (motion in contact) level, whereas section "Randomized path planning applied to (dis)assembly" deals with random path planning methods applied to (dis)assembly. Finally, some summarizing considerations are provided in the "Conclusions".

Part I: Combinatorial aspects of assembly sequencing

Representing assembly sequences

Most representations in assembly sequencing are part- and subassembly-based. Parts, as atomic elements, are represented by labels, symbols, nodes, vector elements and the like. Subassemblies, on the other hand, can be described either by their constituting parts or by the connections established between them, and are represented by nodes, lists, vectors or subgraphs. In this section we present the different ways the space of possible assembly sequences can be represented based on such elements. Alternatively, *features* may be considered instead of parts (in some cases they are coincident), with the possibility of two parts mating with different features pairings [see, e.g., the *Graph of Features* and its use to determine feasible assemblies in Thomas and Torras (1992), the rule-based inference mechanism to determine liaisons between features (Deshmukh et al. 1993), the *connector-based precedence graph* (Tseng and Li 1999), or the *connection semantics based assembly trees* (Dong et al. 2003)].

An *assembly state* is described either by a partition of the set of elemental parts grouped by subassemblies attained so far (symbols representing the parts between braces), or by a binary vector encoding at each digit whether the corresponding connection is established or not. All the representations and sequencing methods described hereafter assume a unique positioning of the part in the assembly.

Assembly sequence representations can be divided into explicit and implicit ones, as suggested in de Mello and Sanderson (1991), where also formal definitions of different representations, correctness and completeness proofs, and mutual relationships between them are given. The input information for obtaining these representations is a set of n parts and the *graph of connections*, whose nodes are these individual parts and the links stand for contact connections between them. This representation is also called *liaison diagram*, although the term *liaison* may have a broader sense than just a contact connection (Fazio and Whitney 1987). A *liaison matrix* can be derived from the connection or liaison graph: each one of the n files or columns corresponds to a part, and the elements of the matrix are equal to 1 if a connection between these two parts exists, 0 otherwise (elements of the diagonal are obviously null) (Lai and Huang 2004). For example, the liaison matrix corresponding to the assembly in Fig. 1 is (rows and columns correspond to the parts **c**, **t**, **b** and **h**, in this order):

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$

The number of connections or of liaisons lies between $n-1$ and $\frac{n(n-1)}{2}$ (Hong and Cho 1999). Augmented versions exist like the *Datum Flow Chain* (Mantripragada and Whitney 1998) that captures dimensional constraints along one or more degrees of freedom between the parts, or the *relational model* (de Mello and Sanderson 1989) by adding information about the type of contact (defined by its geometry), the type of attachment associated to given contacts (glue, screw, pressure fit, etc.), as well as attributes of all the parameters in the assembly.

Explicit representations

In this kind of representations, there is a one-to-one correspondence between assembly operations and the elements of the representation. A single assembly sequence can be represented by a **partial assembly tree**, a binary tree whose nodes correspond to partial assemblies occurring during the execution of the plan, the root node is the final assembly, and the leaves are the single parts. In Wolter (1991), this basic representation is extended with additional information: the *ordering* of operations (by numbering the nodes, this is equivalent to the *state sequence* representation), the *insertion* information (which piece is held on place while the other one is inserted, this is depicted by arrows from the held subassembly to the moved part), or the *fixturing* information, which groups together the subassemblies built in different fixtures before brought together.

As for representations of *sets* of assembly sequences (or of the whole space of feasible sequences), a straightforward and common representation is the **directed graph**, whose nodes are stable partitions of the set of parts, and the arcs correspond to feasible assembly operations. The root node consists of a partition in the N elemental parts, whereas the partition of the leaf node has one single element, the whole

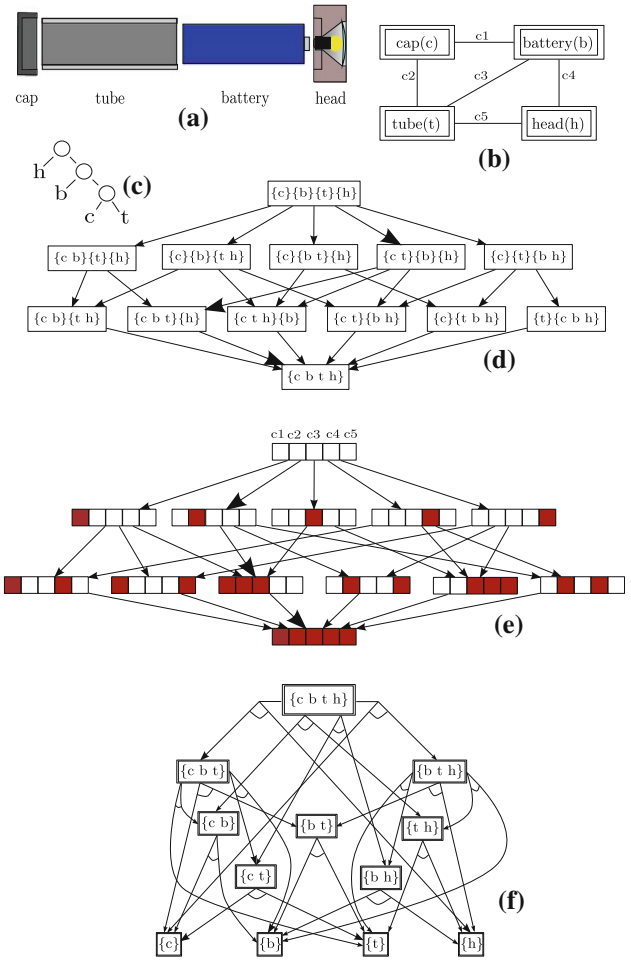

Fig. 1 Different ways of representing the same assembly (**a**), a torch displayed in disassembled form (the head is treated here as a single part, for simplifying reasons). The graph of connections (**b**) is the starting point for deriving representations like the directed graph of assembly states, in its two versions of partitions of parts (**d**) and truth values—unfilled is FALSE, filled is TRUE—of established connections (**e**). A partial assembly tree (**c**) for a specific assembly sequence is also shown. The same sequence is represented with *heavy lines* on the directed graphs and on the AND/OR graph (**f**). Reworked from de Mello and Sanderson (1991), Fazio and Whitney (1987), Wolter (1991)

assembly. Alternatively, the assembly states can be encoded by the truth value of the established connections: the root node would be a sequence of FALSE values, as the parts are initially disconnected, and the leaf node a sequence of TRUEs, corresponding to the final assembled product. This representation is called *state lattice* in Wolter (1991), and used in Fazio and Whitney (1987) (in its alternative formulation). The TMA (*Topological Modelling of Assembly Systems*) in Almgren (1994) can also be seen as a directed graph intended to express not only parts and contacts, but also system components (devices and machines) and locations. Another quite popular representation is the **AND/OR graph**, where nodes are stable subassemblies, and the hyperarcs correspond to feasible assembly (if viewed down-up) operations. Each node is linked to various alternative AND combinations of subassemblies. The root node is the full assembly, and the leaves are the individual parts. AND/OR graphs are used in de Mello and Sanderson (1991), Romney et al. (1995), Wilson and Rit (1990), Lee and Saitou (2003), Thomas et al. (2003).

These representations are illustrated in Fig. 1.

Implicit representations

A necessary step to construct explicit representations of an assembly is to determine the **precedence relationships** between connections, but at the same time they may be considered as a way to implicitly encode feasible assembly sequences. Such partial orderings on the connections are obtained either by considering systematically the relative precedence of specific pairs of connections (Bourjault 1984), or between a given connection and sets of connections (Fazio and Whitney 1987) or assembly states (de Mello and Sanderson 1991) (in the latter case, precedence relations are called **establishment conditions**). These procedures imply asking a human expert, and recently new efforts of incorporating human expertise in the assembly sequencing process to reduce complexity have been considered (Yuan 2002).

Alternatively, precedence relations can be generated from liaison diagrams by the more efficient cut-set method (de Mello and Sanderson 1989). It consists in computing the cut-sets of the diagram, i.e., minimal sets of edges whose removal renders the graph disconnected. This is efficiently done by considering all the subgraphs whose number of nodes is less or equal to the half of the cardinality of the whole graph. For each such subgraph, a cut-set is obtained if the removal of edges of the whole graph that have only one end in the subgraph leaves the original graph with exactly two components (de Mello and Sanderson 1989). Each such cut-set decomposition is tested for its geometric feasibility. Figure 2 shows the cut-sets that correspond to the assembly of Fig. 1.

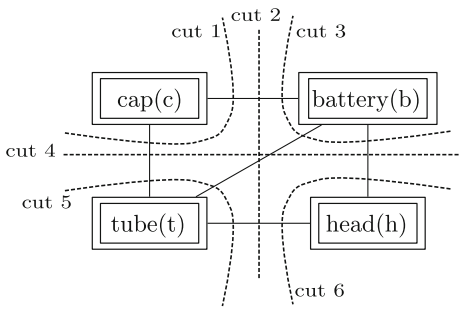

Fig. 2 A graph of connections and its feasible (*1*, *2*, *5*) and non-feasible (*3*, *4*) cut-sets (from de Mello and Sanderson 1989)

Precedence relationships can also be indirectly encoded as **Geometric constraints**, which express the absence of collision-free trajectories that allow two subassemblies to contact. They are used in Bonneville et al. (1995) to detect assembly sequences that require ternary operations (i.e., three-handed operations).

Establishment of connections are called *mating operations* in Wolter et al. (1991). In this work, different types of *mating constraint expressions* are constructed from all possible precedence relations: 'strictly precede', 'precede or accompany', 'same operation', and 'different operation'. These expressions, whether applied locally (to contacts involving a common part), restricted to a given graph, or generally valid, are tested for their representational power (i.e., whether different types of plans like partial assembly trees or state sequences can be exactly represented by them). Other authors (Naphade et al. 1999a) define assembly sequencing as a constraint satisfaction problem in terms of establishment conditions. This 3-SAT problem is decomposed into a collection of 2-SAT problems, which in turn are mapped onto *Decision Graphs* (introducing the concept of *decision dependent constraint*) which are partitioned into self-consistent solutions and rejections. In a companion paper (Naphade et al. 1999b), optimal sequences are computed, according to graph-computable performance measures.

Precedence constraints may affect not only assembly tasks but also the use of resources like fixtures and grasps, and they are called *resource constraints* in this case (Huang and Lee 1991). A generalization of ordinary to AND/OR precedence constraints can be found in Möhring et al. (2004), who provide a linear-time algorithm for deducing additional constraints from existing ones and proving the feasibility of the original set. Also Lai and Huang (2004) use precedence Boolean expressions, not only between parts, but also between feasible and stable subassemblies.

Furthermore, the robots themselves displacing assembly parts and subassemblies pose also accessibility and collision constraints on the assembly process. This problem is addressed in Heger (2008) by validating the edges of an augmented directed acyclic graph that represents the space of feasible assemblies. This validation consists in solving local motion planning problems of the affected components and the robots that carry them.

On the complexity of sequencing

The complexity of assembly or disassembly sequencing is measured generally in terms of the number of parts n, if these parts have a simple geometry (like disks, for example) or the total number of vertices N, when dealing with polygonal or polyhedral parts. However, this measure alone does not express how difficult it is to obtain a valid assembly sequence. Other involved features are the **number of hands** (the maximum number of subassemblies that are moving with respect to one another by any assembly operation), **monotonicity** (whether or not operations of intermediate placement of subassemblies are required), **linearity** (whether all assembly operations involve the insertion of a single part in the rest of the assembly or more than one part have to be simultaneously inserted), and **coherence** (whether or not each part that is inserted will touch some other previously placed part). The simplest sequences are the two-handed or **sequential**, **monotone**, **linear**, and **contact-coherent**. Some of these concepts are illustrated in Fig. 3.

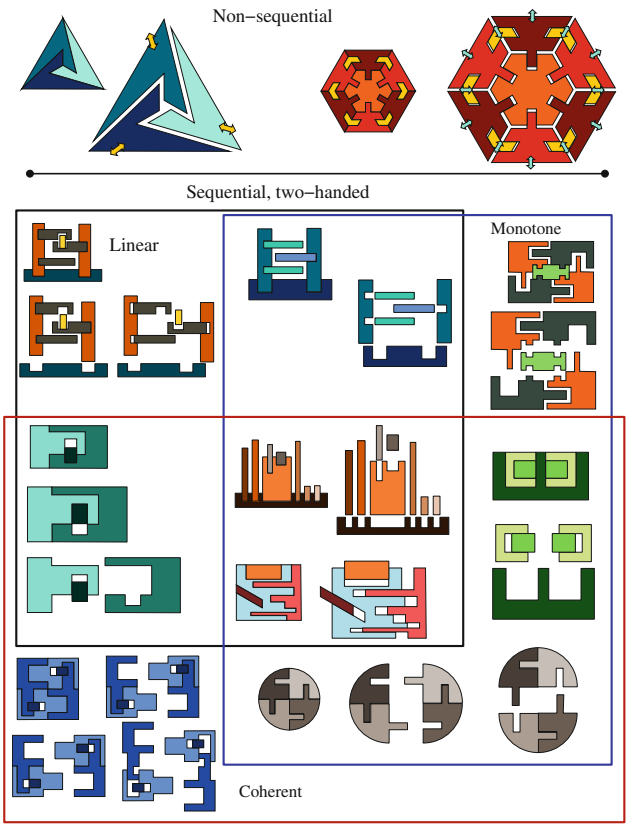

Fig. 3 Different types of assembly sequences (from Wolter 1989, 1991; Romney et al. 1995; Marian 2003)

Optimality criteria, related to the necessary resources of the assembly system (like the required number of degrees of freedom of the robot, or the need of fixations), as pointed out in Goldwasser and Motwani (1999) (see also Chakrabarty and Wolter 1997), constitute valuable tools for evaluating and selecting different sequencing alternatives:

- Number of directions in which parts have to be displaced,
- Number of reorientations or direction changes,
- Number of tools and tool changes,
- Number of non-linear steps,
- Depth of an assembly sequence.

The sequencing algorithms will try to keep these measures as low as possible. The first two have not to be mixed up: consider the pathological example of an assembly with two insertion directions that have to alternate at each step. The third refers to grippers and special devices needed to handle parts and subassemblies that come in different sizes and shapes. The fourth one is related to the cost of displacing (or fixing) whole subassemblies instead of single parts, and the fifth one privileges shallow assembly trees considering that operations can be taken in parallel.

Many other criteria may be defined as well, depending on the particularities of the product. For example, energy consumption reduction may foster sequences where large parts are displaced less, or, for huge products, favor spatial grouping of operations (locality). Efficiency (both in energy needs and in required operation time) does also favor operations where the most involved movements concern easily manipulated parts (manipulability criterion), and the grouping of operations affecting similar parts so as to minimize the needed tool changes (uniformity). The same uniformity criterion favors grouping together categories of assembly operations like screwing, pressing, etc. (Dini et al. 1999). Non-assembly operations like greasing or coating parts may also have to be considered regarding the optimality of assembly sequences. As pointed out in Chakrabarty and Wolter (1997), criteria like safety considerations or taking into account internal mechanical stresses during the assembly process are difficult to be satisfactorily implemented for evaluation purposes in an assembly planner, but—at the same time—plans that do not dare about such issues may become useless. A systematic and comprehensive listing of constraints to be considered in assembly planning can be found in Jones and Wilson (1996).

In the case of partial disassembly (i.e., for recycling a given part), the number of steps to reach the part is also a meaningful measure to be minimized (in assembly or full disassembly, the number of steps for any two-handed sequence will be always the same, $n-1$).

The size of the solution space (i.e., the space of all potential assembly sequences) can give an idea of the complexity of finding an optimal sequence (Marian 2003), as it will increase linearly with this size in the case of exhaustive search. If the number of assembly operations equals the number of parts, the number of potential sequences is given by the number of permutations of parts ($n!$). Of course, non feasible sequences are also included in this amount. Considering permutations means that the sequences are linear and monotone. If the linearity constraint is lifted, the size of the solution space increases to $\frac{(2n-2)!}{(n-1)!}$, and if also non monotone sequences are allowed, the number of potential sequences is, obviously, infinite (Marian 2003).

More restricted and better fitting computations of the real complexity of assembly sequencing do not take just the number of operations, but rather the topology of the precedence graph into account. To this end, Ramos et al. (2001) develop a new representation, the Parse Tree, following the rules of the so-called slot-block theory, on which it is possible to compute the total number of plans given the precedence graph.

Sequencing and optimization algorithms

Determining a feasible assembly sequence or selecting the best one according to some criterion is the result of applying a search and/or optimization algorithm in the space of possible assemblies. The most immediate way is to perform a graph search on the representations described in section "Representing assembly sequences". Exhaustive search is the simplest strategy ensuring completeness but is impractical except for very simple assemblies. Even heuristic graph search strategies have limitations due to combinatorial explosion when the number of parts increases. Several authors have addressed such huge search spaces with search and optimization paradigms of proven efficiency in similar settings.

Graph-search algorithms

In the previous sections directed graphs and AND/OR graphs have appeared as formalisms to encode the space of feasible assembly sequences. Standard graph-search algorithms apply in most cases, with minor modifications, to determine a feasible or an optimal sequence.

The natural choice for directed graphs is an heuristically guided search algorithm like A* (Nilsson 1980), which is guaranteed to find always a minimum-cost path (i.e., assembly sequence). The equivalent algorithm for AND/OR graphs is AO* (Martelli and Montanari 1973, 1978; Nilsson 1980) and all the different versions of it (Bagchi and Mahanti 1983; Mahanti and Bagchi 1985). All these algorithms work for acyclic implicit graphs. Cycles may arise in AND/OR graphs in the assembly/disassembly context, as shown in Jiménez and Torras (2000).

Alternatively to perform a graph search, the liaisons graph may be transformed into a table of liaisons in matrix form (with ones at those elements where a liaison exists between the parts of the corresponding row and column, zero otherwise). A feasible assembly sequence is determined by successively deleting the columns of the parts already included in the assembly and examining their rows for other candidate liaisons to be established (Marian et al. 2006) (see also the PhD Thesis of the first author).

Petri Nets

Petri Nets can encode the information relative to the space of possible assemblies, where the *places* stand for the

different possible subassemblies and the *transitions* correspond to different binary partitions of these subassemblies. Such a construction is described in Zha et al. (1998a), where the feasible subassemblies are obtained from considering topological, geometrical, stability and partial precedence constraints, and also optimality criteria are given for assembly sequence selection and evaluation. In Suzuki et al. (1993), Caselli and Zanichelli (1995), Cao and Sanderson (1998) procedures for building Assembly Petri Nets from AND/OR graphs are described, and the existing body of theoretical results and standard analysis algorithms of Petri Nets is used to derive efficient computations of assembly sequences. A survey on the broader subject of the use of Petri Nets in assembly and task planning can be found in Rosell (2004), with special emphasis on Colored Petri Nets (a type of high level Petri Nets). A knowledge-based Petri net is defined in Zha et al. (1998b) and used in a planning systems which can adjust automatically the deviations between theoretical and real assembly parameters, to guarantee the best plans for flexible assembly. Another hybrid of knowledge-based system and colored Petri Net has also been used to evaluate the degree of difficulty of assembly sequences, which allows to choose the optimal one (Ben-Arieh et al. 2004).

Advanced sequencing and optimization techniques

Due to combinatorial explosion, the previous graph-search methods, even if some heuristic is applied, are impractical except for simple assemblies. Assemblies with 20 parts or more can be approached by using the powerful optimization paradigms developed during the last decades.

Simulated annealing. An energy function is defined in terms of optimality criteria like total assembly time and number of reorientations (Motavalli and Islam 1997) or the normalized degree of motion instability and, again, the number of direction changes (Hong and Cho 1999). In the latter work, precedence and connectivity constraints are also included in the energy function, whereas in the first case precedence constraints are encoded in a precedence diagram and are tested for each time a new sequence is generated. In either case the energy function is minimized following a simulated annealing strategy. The procedure consists in interchanging two arbitrary parts (the initial sequence is also generated randomly), computing the energy corresponding to the new sequence, computing the Boltzmann probability of changing to the new energy state and accepting the new sequence if this probability is larger than a random number in the interval [0, 1] (this is done to escape local minima). The annealing temperature used in the Boltzmann probability computation is a decreasing function of the iteration step number.

Neural networks. Neural Nets (NN) are used to encode the precedence knowledge, by expressing AND/OR precedence constraints between liaisons in the connection strengths between neurons of a Hopfield net (Chen 1992), or the probability of each part to be assembled at each step in the neurons' outputs of an *nxn* network (Hong and Cho 1995). As usual in NN, energy functions can be defined related to the input and output values of the neurons, and in the present case they are formulated so as to correspond to an optimal assembly sequence when a global optimum of these functions is reached. Other more recent works using back-propagation neural networks for assembly sequence optimization are Cem Sinanoglu (2005) and Chen et al. (2008).

Genetic algorithms. Assembly sequences are encoded into chromosomes. Each chromosome corresponds to an individual of a population of feasible assembly sequences. Genetic operators are applied to these chromosomes in order to produce fitter offspring, according to some optimality criteria. Genetic algorithms are executed on an initial population of arbitrary feasible assembly plans, and end—after a number of generations—with a set of good-optimal or near-optimal-plans. The process is stochastic: random choices, with certain fixed probabilities, can be made in the application of the genetic operators (the gene that mutates, the parents selected for crossover) or in the order they are applied. In the pioneering work of Bonneville et al. (1995), for example, crossbreed is systematically applied to the pair of fittest individuals, but mutation is randomly applied on the offspring, while in Sebaaly et al. (1996) reproduction, crossover and mutation are applied in a cyclic fashion but the mates for crossbreeding are selected randomly. Alternatively, Chen and Liu (2001) propose a multi-layered strategy where the genetic operator probability setting itself is updated dynamically in a second-level genetic algorithm whose chromosomes encode the probability of applying each operator in the primary level. This avoids the problem of selecting the adequate mutation rate, which is a compromise between premature convergence to local minima (lower final solution quality) and overall convergence rate (longer run times) (premature convergence is also addressed in Smith (2004) by the use of new genetic operators). An Ordering Genetic Algorithm is developed in De-Lit et al. (2001) which includes a validation mapping of the sequences generated by their genetic operators into feasible ones. In Marian et al. (2006) the scope of types of considered assemblies is substantially enlarged, including also non-sequential, non-linear, non-monotone, and pseudo-non-coherent assembly plans, by encoding into the chromosomes not just plain sequences of parts: genes are what the authors call *Entities Meaningful for the Assembly Sequence* (EMAS), which include single parts, whole subassemblies, sets of parts to be simultaneously positioned, operations without the addition of a new part, auxiliary fixtures, etc. The initial population of feasible assemblies is generated by guided search in the table of liaisons, as described at the end of section"Graph-search algorithms". This table encodes intrinsic precedence constraints, the extrinsic ones (derived from accessibility and

process constraints) are used to select among the candidate liaisons. As for the optimization process, both in the crossover and pseudo-mutation operators guided search is used again to guarantee the feasibility of the resulting chromosomes. Genetic algorithms are also used in Guan et al. (2002) for solving the broader problem of assembly process planning, including into the chromosome information like the assembly direction, the tool to be used, or the type of assembly operation. Similarly, Tseng et al. (2004), Wang and Tseng (2009) encoded *connector-based* information into the chromosomes, where connectors are features acting as assembly elements (this provides a more engineering-like description of assemblies, while at the same time reducing the combinatorial complexity).

Other biological analogues

Closely related to GA, another population-based optimization technique, called Immune Optimization Approach (IOA) uses the bionic principles of Artificial Immune Systems (AIS) (Cao and Xiao 2007). The assembly sequencing problems are represented as antigens, and antibodies represent the assembly sequences of the product, encoded in their genes as component numbers. The authors claim that IOA performs better as standard GA due to the *immune selection mechanism*, which selects individuals (antibodies) for the next generation, choosing the best (higher fitness value) antibodies while at the same time favoring diversity, i.e., avoiding premature convergence and helping global optimization.

Ant Colony Optimization (ACO) is a meta-heuristic method inspired in the cooperative behavior of ant colonies in finding the shortest path to the food source, combining random movements with reinforcement of specific trails with pheromone traces. It has applied to assembly-by-disassembly sequence planning in Wang et al. (2005). A *Disassembly Completed Graph* represents the search space of all possible disassembly sequences (where each node corresponds to a part and a disassembly direction, i.e., to an elemental disassembly operation DO), planning consists in finding a path that joins nodes with different part identification numbers, respecting the geometric precedence constraints. Local pheromone updating is done by each ant going from one DO to an adjacent one, which encourages exploration of alternative solutions, while global pheromone updating, consisting in evaporation of pheromone in all edges and extra addition of pheromone to the trails with least reorientations, enforces exploitation of the most promising solutions. In Cui (2007) an adaptive ant colony algorithm is described for generating optimal assembly sequences of large space truss structures.

Case studies in the literature

Most of these references illustrate their work by providing examples of their algorithms dealing with specific assemblies. There are academic toy assemblies (a 5-part 2D assembly in Jiménez and Torras 2000 and Ben-Arieh et al. 2004, or a 20-part hypothetical product in Motavalli and Islam 1997), but the majority prefer to show experiments on real assemblies. These range from a 4-part pince in Cem Sinanoglu (2005) to the 48-part gear-box in Chen and Liu (2001). Grouped by type of algorithm (the number of parts shown in parentheses), these examples include:

1. **Petri Nets** a flashlight (4) (Caselli and Zanichelli 1995) and a ball-point pen (5) (Suzuki et al. 1993);
2. **simulated annealing** a relay (10) (Hong and Cho 1999);
3. **neural networks** a pince (4), a hinge (4) and a coupling system (7) (Cem Sinanoglu 2005), a gearbox (10) (Chen 1992), a relay (10) and an alternator (13) (Hong and Cho 1995), and an electric torch (16) (Chen et al. 2008);
4. **genetic algorithms** an oil pump (5) (Bonneville et al. 1995), an industrial controller (19) (Guan et al. 2002), an hydraulic linear motor (25) (Marian et al. 2006), an air condition control (28) (Sebaaly et al. 1996), a signaling relay (34) (De-Lit et al. 2001), and the already mentioned gear-box (48) (Chen and Liu 2001);
5. **immune optimization approach** a controller (19) (Cao and Xiao 2007);
6. **ant colony optimization** an industrial driver (16) (Wang et al. 2005).

This list—which by no means pretends to be exhaustive—should be interpreted as an illustration of the variety of assemblies dealt with in the existing literature, not as a ranking of the suitability of the different sequencing and optimization algorithms. Furthermore, the examples provided by the authors are generally more oriented towards explaining how their algorithms work than to demonstrate their performance. Benchmark assemblies, both theoretical (puzzle-like examples have already been proposed in Le et al. 2009) as well as real ones could be defined, incorporating the means to evaluate different optimality criteria. Such standard test assemblies would allow consistent comparative studies, and would constitute a valuable tool for researchers for analyzing and validating their algorithms.

Part II: Geometrical issues

Up to now, it was assumed that the knowledge about precedence of connections was available, or obtained by systematically posing to a (human) user questions like those of Bourjault (1984). Alternatively, they can be learned from a human demonstrator by an automatic learning system (Kuniyoshi et al. 1994). However, it would be desirable to

obtain this information direct- and automatically from a CAD description of the assembly. Early works pointed towards a generate-and-test approach (Lee and Shin 1990; Wilson and Rit 1990; Deshmukh et al. 1993), but even if mechanisms for saving and reusing previous results are provided, the inherent combinatorial explosion makes this approach impractical except for very simple assemblies.

Three lines of action have emerged along the years. All three of them rely on the primary information provided by the CAD model, which refers to the contacts between the parts' boundary features. They differ in how this information is used. In broad words, the first line uses the directional information related to the contacts to determine blocking relationships between parts, the second one explores the space of feasible contacts, combinations and transitions between them, whereas the third exploits geometry to bias random path planning methods. These lines are schematically summarized in Fig. 9.

Separability and blocking relationships between parts

Representation: DGBs and NDBGs

Following the assembly-by-disassembly principle, precedence is equivalent to blocking relationships between parts. Thus, the problem can be stated as determining the directions along which specific parts in contact may be separated from one another. The contacts between boundary features bound the ranges of directions along which it is possible to separate the contacting parts in an assembly. Such partitions of the space of separating directions are computed assuming that the parts are free-flying rigid objects (without considering grasping and stability issues). Furthermore, in most cases "infinite" separating translational motions are assumed, but other displacements like infinitesimal translations in 2D and 3D or infinitesimal generalized motions (i.e., including rotations) can be considered as well (see Wilson and Latombe 1994 for a description of the corresponding representation spaces and the construction complexities). In the simple planar case, edge-edge contacts induce a partition of the circle of directions $\mathcal{S}^1$ (Wilson and Latombe 1994). As for 3D assemblies, point-plane constraints between two polyhedra determine the sets of allowable (infinitesimal) motion directions, as closed hemispheres on the unit sphere $\mathcal{S}^5$ in six-dimensional space (Guibas et al. 1995). The authors provide an algorithm that computes representative separating motion directions by characterizing *maximally covered cells* induced by the partition of $\mathcal{S}^5$ by the great circles that limit the hemispheres (actually on a central projection of this partition on a tangent hyperplane). Similarly, *local translational freedom cones* are computed on $\mathcal{S}^2$ (whose apices are on the center of the sphere) for translating polyhedra, in Romney et al. (1995), Romney (1997), or *local depart spaces* as three-dimensional polyhedral convex cones in Mosemann et al. (2000). Contacting face normals can also be used in the context of randomized path planning, as explained in section "Randomized path planning applied to (dis)assembly" below. Without entering in the planning process, a more general local characterization of infinitesimal separating motions is proposed in Staffetti et al. (1999b). It applies to general polyhedra (i.e., possibly with non-convex faces), relying only on *basic contacts* (see section "From geometry to compliance: (dis)assembly as motion in contact"). The fact that the contact relations between parts can be expressed in form of linear constraints, as hyperplanes embedded in the assembly configuration space (Schweikard and Schwarzer 1998), is used in Schwarzer et al. (2000) to determine feasible translational directions for m-handed disassembly operations.

The resulting regions where the mutual blocking relationships remain constant can be labeled with *Directional Blocking Graphs* (DBG), defined for a representative direction inside each region: nodes correspond to individual parts, and an arc points from P_i to P_j iff P_j blocks the displacement of P_i along this particular direction (this is the same concept as the *Local Constraint Digraph* in Lu et al. 1993). Enhanced variants of the DBG exist, like the *Directional Force Graph*, where the links between the parts are labeled with the maximum static force that has to be exerted to achieve an infinitesimal displacement between these parts along a specific direction (Lee and Moradi 1999). The partition of the sphere of directions labeled with the corresponding DBGs is called the *non-directional blocking graph* (NDBG). Figure 4 displays the NDBG for an assembly in a 2D infinitesimal translational case.

Algorithms: computing partitionings and disassembly motions

The notion of NDBG was introduced in Wilson and Latombe (1994), where it was used to compute candidate partitionings of assemblies. A *strong component* of a DBG (in general, of any di-graph) is a maximal subassembly (set of nodes) such that for any pair of parts (nodes) a path exists connecting them. If only one strong component exists, there are no possible partitions of the assembly (the DBG is strongly connected). Thus, a disassembly algorithm tries to identify subsets of the DBG without outgoing arcs. This method applies to disassemblies constituted by one-step motions. The authors developed later the *interference diagram* (Wilson et al. 1995) which allows to determine a multi-step collision-free path for a subassembly as a sequence of connected cells in this representation. It is obtained by superimposing the Minkowski differences of all pairs of parts in the assembly, computed with respect to the same reference point. The resulting cells are labeled with the respective colliding parts. The reference point is contained in the initial

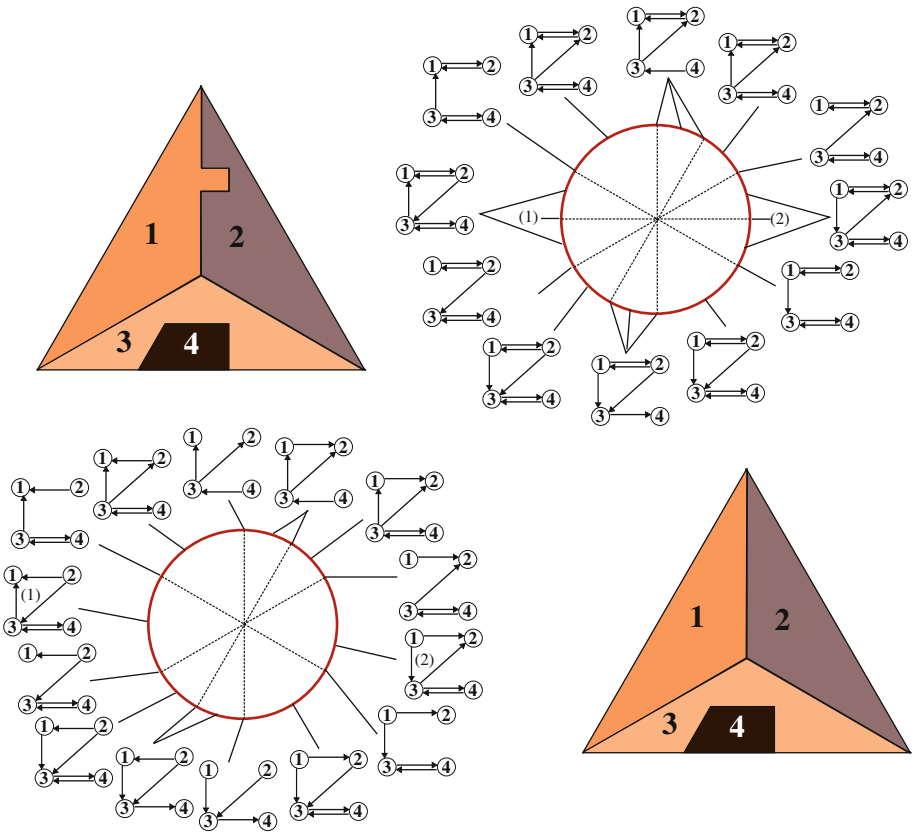

Fig. 4 Two assemblies and their respective NDBGs. A slight but significant difference between the upper and the lower assembly alters not only the partition of the circle of directions, but also the DBGs attached to each region. Elaborated on Wilson and Latombe (1994)

cell, and the goal is to reach the outermost final cell. At each traversed cell along a path, the corresponding constraints are added to the blocking graph. If the DBG becomes strongly connected by traversing a particular cell, an alternate route has to be determined. Figure 5 shows an assembly of two parts, the corresponding interference diagram, and the multi-step paths for disassembly.

Both methods are presented again in Halperin et al. (2000) under the unifying notion of *motion space*, defined as "the space of parametric representations of all allowable motions for partitioning operations". One-step translations in 3D, for instance, can be represented on a 2-dimensional motion space, as only two parameters are enough to define the direction of motion. A common practice is to refer the motions of the parts with respect to a universal frame (a given point in assembled state), so that the parametrization is independent of the particular subassemblies to be displaced. A similar concept is presented in Thomas et al. (2003), where separability directions are generated and evaluated by the computation of the Minkowski differences of each pair of parts in the assembly and stereographic projections of the C-obstacles obtained this way. AND-conjunction can be applied to the binarized images of these projections, where value 1 stands for the maximal distance form the origin to the C-obstacles, along the AND/OR tree corresponding to the assembly: subassemblies with remaining 1-values are geometrically feasible (i.e., separable).

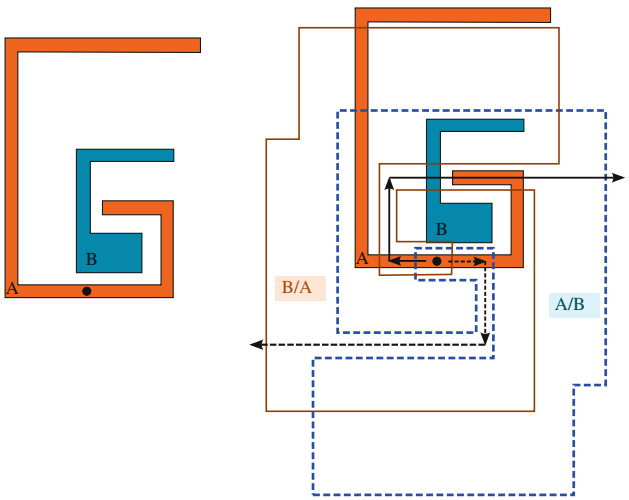

Fig. 5 Interference diagram for an assembly consisting in two pieces. The *solid arrows* display a disassembly path for *B* while maintaining it A on place (the reference point—the *black dot*—has to be seen as rigidly attached to part *B*). Similar- and symmetrically, *dashed arrows* correspond to a multi-step disassembly path for part *A*, as they only traverse cells labelled with *B/A*. Inspired in Wilson et al. (1995), which shows an example involving three parts

From geometry to compliance: (dis)assembly as motion in contact

Representation: the space of contact states

Motion in contact constitutes a whole category in robot motion planning (which deals with control issues, uncertainty management, etc.) and has an evident link to assembly planning, where the most relevant point is to identify the possible contact states, the degrees of freedom associated to them, and how to go from one specific contact state to another. This information is extracted from the geometrical models of the parts and the assembly, and used to construct a graph representation suitable for path planning. In the case of polyhedral parts, this involves the following steps:

- A set of **elementary contacts** between boundary primitives is defined: In Donald (1985) and Dakin (1994) only two such *basic* or *primitive* (respectively) contacts between convex features are considered [vertex-face (v-f) and edge–edge (e-e)], whereas all ten possible combinations (see Fig. 6) constitute the set of *canonical* (Giraud and Sidobre 1992) or *principal* (Xiao 1993; Xiao and Ji 2000, 2001) contacts (although in Meeussen et al. 2004 each principal contact is further decomposed in the two elementary contacts mentioned above). The inclusion of higher-level primitives in these sets aims at achieving higher robustness to uncertainties (Xiao and Liu 1998).
- These elementary contacts constitute the building blocks of **polyhedral contacts**, defined as *lists of canonical contacts* at a specific configuration, in Giraud and Sidobre (1992), or as *contact formations*, CF, in Desai et al. (1988), Xiao (1993), Xiao and Ji (2000, 2001). These sets correspond to the nodes of the graph where the assembly sequence is to be planned. In order to establish links between them, a neighborhood relationship has to be defined.
- **Connectivity** between neighboring contacts (e.g., CF_i and CF_j) is provided by the existence of a contact motion that leads from CF_i to CF_j and which does not include any other contact (Dakin 1994) (see also Fig. 7). Critical configurations (where the *local contact space* changes) are identified along a given nominal trajectory computed by a high-level planner (which does not take uncertainties into account). Thus, connectivity is already implicitly provided, as an adjacency graph of contact states. Alternatively, both in Giraud and Sidobre (1992) and in Xiao (1993); Xiao and Ji (2001) candidate neighbors are looked for by eliminating elementary contacts from the lists that define the polyhedral contacts. In the first reference, however, this relaxation procedure is restricted to determine those neighbors that have only one degree of freedom, whereas in Xiao and Ji (2001) relaxation is applied iteratively to a set of highly constrained seed CFs obtaining progressively less and less constrained neighbors and building up the so-called *Goal-Contact Relaxation* (GCR) graphs. These graphs are merged together to construct the contact formation graph. In any case, feasibility of infinitesimal translations and rotations or finite translations is checked for, in order to eliminate non-admissible (i.e., colliding) neighbors.

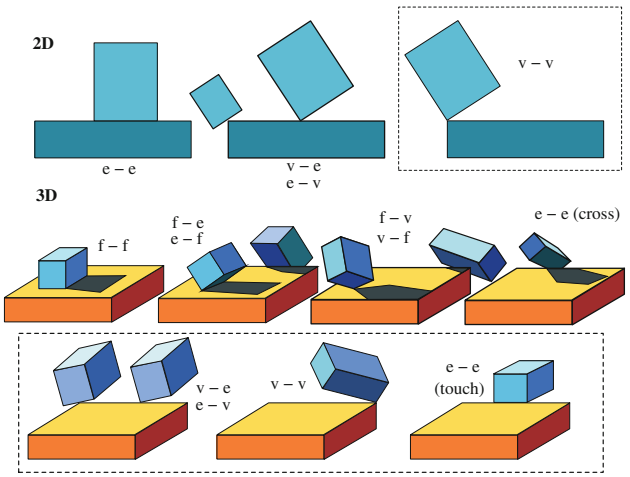

Fig. 6 Principal contacts in 2D and 3D. Remark that v-f is different from f-v, and the same applies to e-f/f-e and e-v/v-e. Furthermore, two possible e-e exist depending on whether they cross at one point or are aligned. Basic contacts are also displayed in *bold lines*. Degenerate contacts are enclosed in *dashed rectangles*. From Xiao and Ji (2001)

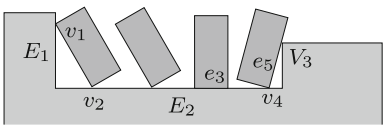

Fig. 7 Transitions between contact formations. From *left* to *right*: a small displacement (a small rotation would produce the same effect) transforms the contact formation $\{v_1 - E_1, v_2 - E_2\}$ into $\{v_1 - E_1\}$, an additional rotation leads to $\{e_3 - E_2\}$ and a final displacement plus rotaion to $\{v_4 - E_2, e_5 - V_3\}$

Algorithms: planning in the space of contact states

Two planning levels may be distinguished: a higher symbolic level on the graph of contact formations in search of an optimal sequence of contact states, and a lower contact compliant motion planning level (Xiao and Ji 2000; Bruyninckx et al. 2001). It is the combination of these two levels which finally results in a sequence of executable commands by the robot.

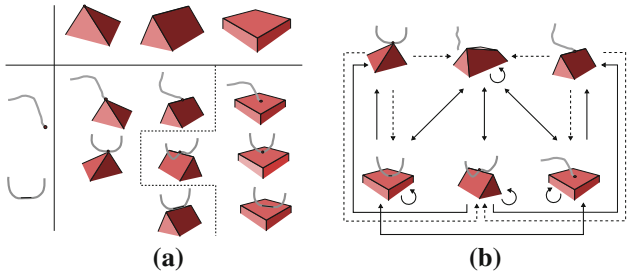

Fig. 8 **a** Contact states between the features of a deformable linear object (vertex and edge) and the features of a polyhedron (vertex, edge, face), after Remde et al. (1999). The *dotted line* separates stable (on the *right*) from unstable states. Punctual and linear edge/edge and edge/face contacts are distinguished. **b** In the state transition graph, the vertex/vertex and linear edge/edge contacts are not considered, as they are unlikely to occur as initial contact states, and the punctual and linear edge-face contacts are considered together. The non-contact state is represented (*top center*). *Solid links* indicate reversible transitions: the outcome can be ensured by a controlled manipulation. Transitions starting at unstable states have several possible stable successors and are shown in *dashed lines*

High-level planning

A heuristic graph search can be applied to contact space representations, like in Giraud and Sidobre (1992), where planning in the *contact graph* is guided by simple geometric rules to select one of the 1-dimensional neighbors at each node. Similarly, a sequence of recognizable (by force sensors, e.g.) contact state transitions is obtained in Dakin (1994) with an heuristic graph search. Contact states and the possible transitions between them have later been studied in the context of articulated bodies (Staffetti et al. 2005), as well as of the manipulation of deformable linear objects (Remde et al. 1999, 2000).

A similar framework has been developed beyond the rigid body case: the different possible contact states between a linear deformable object and a rigid polyhedral body are identified in Remde et al. (1999), and the feasible transitions between these states are listed (see Fig. 8). A further elaboration on this formalism characterizes contact states by their stability and defining contact state transition classes (Acker and Henrich 2005).

Low-level contact compliant motion planning

A strategy which consists in restricting the search of potential removal directions to those which are perpendicular to contacting part faces can be applied to perform a motion while maintaining a specific contact formation, as shown in Ji and Xiao (2001a,b), (for general *infinitesimal* motions maintaining given sets of contacts, see Staffetti et al. 1999a). In Ji and Xiao (2001b) algorithms are described that generate random samples of configurations compliant with CFs consisting of one or two principal contacts (PC). A *Direct Calculation* method can be employed when the ranges of independent variables are easily computable: for example, the translational part of a random displacement that keeps a single-PC CF consists in picking up randomly one point in the interior of each feature—the vertex itself, if this is the feature—and making them coincide (see Ji and Xiao 2001b for the rotational part). In the case of two-PC CFs with two translational and or any number of rotational degrees of freedom a *Hybrid Method* is applied that combines Direct Calculation with resampling or convergent iteration. Further steps are taken in Ji and Xiao (2001a) by providing the means to check whether a CF-compliant configuration is also feasible, i.e., without any other collision, as well as for performing *compliant interpolation* between two feasible CF-compliant configurations.

This work, together with the contact state graph generation (Xiao and Ji 2001), is presented in the context of a general framework that structures all the necessary modules for assembly planning, i.e., modelling, planning, estimation/monitoring, control and task coordination, in Bruyninckx et al. (2001). See also Lefebvre et al. (2005b) for a survey on the state-of-the-art and integration of these modules in active compliant motion. More specifically, Meeussen et al. (2005) presents an approach to automatically generate a task specification for a hybrid controller (the *Compliant Task Generator*) from the output provided by the compliant planner, estimation/monitoring is covered in Mihaylova et al. (2001), and in Meeussen et al. (2004) the feasibility of contact states is restricted further by considering the constraints imposed by the part handling manipulator. The necessary sensing actions themselves can be planned automatically, which is known as *active sensing*, and Lefebvre et al. (2003), Lefebvre et al. (2005a) provide the means to derive an optimal compliant motion task plan consisting of both the sequence of contact formations and the compliant path to be executed while maintaining each CF. Programming a robot to perform a compliant motion task can also be done following the *programming by human demonstration* paradigm, as long as the motion performed by the human demonstrator can be segmented into different CFs while registering the necessary geometric information, as done in Meeussen et al. (2007).

Flexible parts introduce an additional degree of complexity. The necessary corrective motions when misalignments have been detected cannot be derived analytically in general, except for simple settings like flexible beam insertion (Zheng et al. 1991), and thus alternative methods based on training Neural Networks have been devised (Kim and Cho 2000). Contact state transitions for deformable linear objects are detected both with vision (Abegg et al. 1999) and force (Remde et al. 2000).

Randomized path planning applied to (dis)assembly

Representation: composite configuration space

The configuration space (C-space for short) of any solid (in particular, of a given part in an assembly) is defined by all the possible poses or values of the degrees of freedom (variables that define the position and orientation) of this solid. The presence of other solids in this space originates the configuration space obstacles (C-obstacles), i.e., those subsets of C-space where the solid is colliding with these other solids. Path planning consists in determining a path from a start to a goal configuration which is entirely contained in C-free (= C-space − C-obstacle) (Latombe 1991). In the context of an assembly, we may consider the C-space of a given part, and other parts of the assembly originate the C-obstacles of this part. C-free is given by the collision-free space surrounding the assembly plus the tolerances that may exist between the parts. If all the C-spaces of the parts in an assembly are considered simultaneously (where each spatial variable of each part adds one extra dimension) we obtain the composite configuration space of the assembly.

Algorithms: biased randomized path planning

Randomized methods have become hugely popular in path planning, as they provide the means to efficiently tackle high dimensional settings, like the composite configuration space of the parts of an assembly. They are based on randomly choosing a configuration and testing whether it belongs to C-free (collision test). Very simple local planners are used to try to link neighboring free configurations and a roadmap is built in this way. Simple graph search can than be performed on this roadmap to find the solution path (see LaValle 2006; Choset et al. 2005 for more details). As pointed out in Sundaram et al. (2001), (dis)assembly planning involves a repeated presence of narrow passages in C-space, which renders the direct application of probabilistic roadmap methods (PRM) impractical. However, the Iterative Manhattan-like RRT in Le et al. (2009), which performs simultaneous path planning and (dis)assembly sequencing and relies only on collision detection, seems to provide evidence on the contrary. Nonetheless, these same authors point out that in specific (e.g. polyhedral) domains, there is place for improvement by integrating geometric information that gives suitable motion directions. Indeed, the geometrical information attached to the specific contact states can be used to bias the sampling appropriately. This is done in Sundaram et al. (2001) in the context of randomized path planning by restricting the search to potential removal directions, which are perpendicular to contacting part faces.

Just for concluding these sections devoted to the geometry-related aspects of assembly sequencing, Fig. 9 summarizes the alternative paths leading from the pure geometric information related to the contacts between the parts' features to a final feasible assembly sequence.

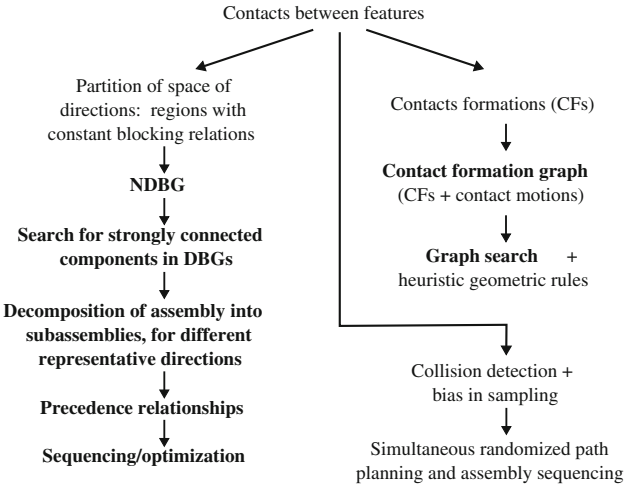

Fig. 9 The three main alternatives to come up with an assembly plan starting from a pure geometrical description of the assembly. Geometrical information and techniques are displayed in plain text, whereas graph search representations and tools are shown in *boldface*

Conclusions

Assembly sequencing is present along the whole lifetime of a product, up from its very conception. It plays a fundamental role both for the assembly as a product and for the assembly as a process. The existence of a feasible sequence confirms that the product can actually be assembled. The computation of such a sequence provides, at symbolic level, an ordering of assembly operations for the manufacturing system. Optimality criteria can be considered to obtain a sequence which not only is feasible but also makes the best possible use of the available manufacturing resources, be it time, energy, cost or whatever. Sequencing is the backbone of the broader problem of planning, where the whole assembly process together with parts feeding, fixturing and transfer systems, etc. is pondered. *Design for Assembly* has emerged as a discipline that integrates design and assembly planning, aiming at enhancing production efficiency and thus at reducing manufacturing costs. The design phase benefits from the achievements in virtual and augmented reality in the last decades, oriented to assembly simulation (see for example Raghavan et al. 1999 and Ji et al. 2002). During the use phase of the product, assembly sequencing is important for maintenance and repair, besides the fact that many products have to be assembled by the end user, like toy models or the furniture of a well-known Swedish company. At the end of the product's

lifetime, recycling and proper disposal imposes also specific criteria on the (dis)assembly sequence.

Assembly sequencing is obviously related to task level planning: Assembly parts or states, as well as precedence constraints and building operations, are symbolically represented. Sequencing is formulated as a combinatorial problem: from all the possible combinations of part symbols, representing temporal orderings of assembly operations, determine those that respect contact and precedence constraints (feasibility) or meet some optimality conditions. This survey has displayed the different ways of representing the space of assembly sequences and the algorithms to determine feasible and/or optimal ones, like graph search techniques, Petri Nets, simulated annealing, neural networks, genetic algorithms, and others. All these methods assume an implicit encoding of feasibility or the availability of (efficient) feasibility tests. Nonetheless, the geometric grounding of this symbolic-level planning has also been addressed in this survey, by describing the geometrical notion of separability which is exploited to derive precedence relationships in an automatic way. Furthermore, the actual execution of (dis)assembly operations, tackled as motion in contact planning, is also covered.

In fact, assembly planning could theoretically be tackled as a pure path planning problem, operating at geometric level. However, the complexity of the assembly problem posed in path planning terms renders it quite (if not too) challenging to solve as the number of parts increases. Random sampling methods may apply successfully if the range of possible (dis)assembly motion directions is not too tight, as shown in Le et al. (2009). Otherwise, the task level domain may provide a plan which can be conveniently translated into a sequence of contact motions. Further constraints concerning grasping and collision avoidance of the robot with the environment can be considered to come up with an assembly plan in terms of robot motion commands. A symbiotic relationship between task-level assembly sequencing and motion planning is established in Heger (2008), where the accent is put on generating robust plans: the edges of a directed assembly graph (whose construction is given by the application of the assembly-by-disassembly principle) are validated by solving the specific motion planning problems associated to them (probabilistic roadmap planners are used to this end in this reference). Most recent and future works point in this direction, at tight integration of the task level with path (and motion) planning, as the output of such systems is practically directly translatable into robot instructions.

Another trend consists in assembly sequencing and planning with flexible parts. Deformation of these parts has to be taken into account when deriving contact and precedence constraints. We have already mentioned some works on motion in contact with flexible parts, but a true integration into an assembly planner is still open research.

Acknowledgments This work has been partially supported by the European project IntellAct (FP7-269959), by the Spanish Ministry of Science and Innovation under project PAU DPI2008-06022, and by the Catalan Research Commission through the Consolidated Robotics Group.

References

Abegg, F., Engel, D., & Wörn, H. (1999). Manipulating deformable linear objects—vision-based recognition of contact state transitions. In: *The Ninth international conference on advanced robotics* (pp. 135–140).

Acker, J., & Henrich, D. (2005). Manipulation of deformable linear objects: From geometric model towards program generation. In: *Proceedings of the IEEE international conference on robotics and automation* (pp. 1541–1547).

Almgren, R. (1994). Comparative topological modeling and analysis of assemblies and assembly systems—an aid in computerized assembly planning. In: *Proceedings of the IEEE international conference on robotics and automation* (ICRA) (pp. 1468–1475).

Bagchi, A., & Mahanti, A. (1983). Admissible heuristic search in and/or graphs. *Theoretical Computer Science, 24*, 207–219.

Ben-Arieh, D., Kumar, R. R., & Tiwari, M. K. (2004). Analysis of assembly operations difficulty using enhanced expert high-level colored fuzzy petri net model. *Robotics and Computer-Integrated Manufacturing, 20*(5), 385–403. doi:10.1016/j.rcim.2004.03.002.

Bonneville, F., Henrioud, J., & Bourjault, A. (1995). Generation of assembly sequences with ternary operations. *IEEE international symposium on assembly and aask planning* (pp. 245–249). doi:10.1109/ISATP.1995.518778.

Bourjault, A. (1984). *Contribution a une approche methodologique de l'assemblage automatise: Elaboration automatique des sequences operatoires*. PhD thesis, Universite de Franche-Compte.

Bruyninckx, H., Lefebvre, T., Mihaylova, L., Staffetti, E., Schutter, J. D., & Xiao J. (2001) A roadmap on autonomous robotic assembly. In: *IEEE proceedings of the international symposium on assembly and task planning* (pp. 49–54). doi:10.1109/ISATP.2001.928965.

Cao, P. B., & Xiao, R. B. (2007). Assembly planning using a novel immune approach. *The International Journal of Advanced Manufacturing Technology, 31*(7–8), 770–782. doi:10.1007/s00170-005-0235-2.

Cao, T., & Sanderson, A. C. (1998). And/or net representation for robotic task sequence planning. *IEEE Transaction on Systems, Man and Cybernetics, Part C Appliccations and Reviews, 28*(8), 104–218.

Caselli, S., & Zanichelli, F. (1995). On assembly sequence planning using Petri Nets. In: *IEEE proceedings of the international symposium on assembly and task planning* (pp. 239–244).

Cem Sinanoglu, H. R. B. (2005). An assembly sequence-planning system for mechanical parts using neural network. *Assembly Automation, 25*(1), 38–52. doi:10.1108/01445150510578996.

Chakrabarty, S., & Wolter, J. (1997). A structure-oriented approach to assembly sequence planning. *IEEE Transactions on Robotics and Automation, 13*(1), 14–29. doi:10.1109/70.554344.

Chen, C. L. P. (1992). Design of a real-time and/or assembly scheduler on an optimization neural network. *Journal of Intelligent Manufacturing, 3*(4), 251–261. doi:10.1007/BF01473902.

Chen, S. F., & Liu, Y. J. (2001). The application of multi-level genetic algorithms in assembly planning. *Journal of Industrial Technology, 17*(4), 1–9.

Chen, W. C., Tai, P. H., Deng, W. J., & Hsieh, L. F. (2008). A three-stage integrated approach for assembly sequence planning using

neural networks. *Expert Systems with Applications, 34*(3), 1777–1786. doi:10.1016/j.eswa.2007.01.034.

Choset, H., Lynch, K. M., Hutchinson, S., Kantor, G. A., Burgard, W., Kavraki, L. E., & Thrun, S. (2005). *Principles of robot motion: Theory, algorithms, and implementations*. Cambridge, MA: MIT Press.

Cui, J. G. P. W. N. (2007). Adaptive ant colony algorithm for on-orbit assembly planning. In: *Proceedings of the 2nd IEEE conference on industrial electronics and applications* (pp. 1590–1593). doi:10.1109/ICIEA.2007.43186.76.

Dakin, G. A. (1994). *Fine-motion planning for robotic assembly in local contact space*. PhD thesis, University of Massachussets Amherst.

De-Lit, P., Latinne, P., Rekiek, B., & Delchambre, A. (2001). Assembly planning with an ordering genetic algorithm. *International Journal of Production Research, 39*(16), 3623–3640. doi:10.1080/00207540110056135-005-0235-2.

de Mello, L. S. H., & Sanderson, A. C. (1989) A correct and complete algorithm for the generation of mechanical assembly sequences. In: *Proceedings IEEE international conference on robotics and automation* (ICRA), (vol. 1, pp. 56–61). doi:10.1109/ROBOT.1989.99967.

de Mello, L. S. H., & Sanderson, A. C. (1991). Representations of mechanical assembly sequences. *IEEE Transactions on Robotic and Automation, 7*(2), 211–227. doi:10.1109/70.75904.

Desai, R., Xiao, J., & Volz, R. (1988). Contact formations and design constraints: A new basis for the automatic generation of robot programs. In: *NATO ARW: CAD based programming for sensory based robots*, (pp. 361–395).

Deshmukh, A., Yung, J. P., & Wang, H. P. (1993). Automated generation of assembly sequence based on geometric and functional reasoning. *Journal of Intelligent Manufacturing, 4*(4), 269–284. doi:10.1007/BF00124140.

Dini, G., Failli, F., Lazzerini, B., & Marcelloni, F. (1999). Generation of optimized assembly sequences using genetic algorithms. *Annals of the CIRP*, *48*, 17–20.

Donald, B. R. (1985). On motion planning with six degrees of freedom: Solving the intersection problems in configuration space. In: *Proceedings of the IEEE international conference on robotics and automation* (pp. 536–541).

Dong, T., Tong, R., Dong, J., & Zhang, L. (2003). Knowledge-based assembly sequence planning system. In: *Proceedings of the 8th international conference on computer supported cooperative work in design* (pp. 516–521).

Fazio, T. L. D., & Whitney, D. E. (1987). Simplified generation of all mechanical assembly sequences. *IEEE Journal of Robotics and Automation , RA-3*(6), 640–658.

Giraud, A., & Sidobre, D. (1992). A heuristic motion planner using contact for assembly. In: *Proceedings of the IEEE international conference on robotics and automation* (ICRA) (pp. 2165–2170).

Goldwasser, M. H., & Motwani, R. (1999). Complexity measures for assembly sequences. *International Journal of Computational Geometry and Application, 9*(4/5), 371–417.

Guan, Q., Liu, J. H., & Zhong, Y. H. (2002). A concurrent hierarchical evolution approach to assembly process planning. *International Journal of Production Research, 40*(14), 3357–3374. doi:10.1080/00207540210146152.

Guibas, L. J., Halperin, D., Hirukawa, H., Latombe, J. C., & Wilson, R. H. (1995). A simple and efficient procedure for polyhedral assembly partitioning under infinitesimal motions. In: *Proceedings of the IEEE international conference on robotics and automation*) (pp. 2553–2560).

Halperin, D., Latombe, J. C., & Wilson, R. (2000). A general framework for assembly planning: The motion space approach. *Algorithmica, 26*, 577–601. doi:10.1007/s004539910025.

Heger, F. W. (2008). Generating robust assembly plans in constrained environments. In: *Proceedings of the IEEE international conference on robotics and automation* (ICRA) (pp. 4068–4073).

Hong, D., & Cho, H. (1995). A neural-network-based computational scheme for generating optimized robotic assembly sequences. *Engineering Applications of Artificial Intelligence, 8*(2), 129–145.

Hong, D., & Cho, H. (1999). Generation of robotic assembly sequences using a simulated annealing. In: *Proceedings of the 1999 IEEE/RSJ international conference on intelligent robots and systems* (pp. 1247–1252).

Huang, Y., & Lee, C. (1991) A framework of knowledge-based assembly planning. In: *IEEE international conference on robotics and automation*, (vol. 1, pp. 599–604). doi:10.1109/ROBOT.1991.131647.

Ji, P., Choi, A. C., & Tu, L. (2002). Vdas: A virtual design and assembly system in a virtual reality environment. *Assembly Automation, 22*(4), 337–342.

Ji, X., & Xiao, J. (2001a). Planning motions compliant to complex contact states. *International Journal of Robotic Research, 20*(6), 446–465.

Ji, X., & Xiao, J. (2001b). Random sampling of contact configurations in two-pc contact formations. In K. Lynch, B. R. Donald, & D. Rus (Eds.), *New directions in algorithmic and computational robotics*. Boston: Kluwer.

Jiménez, P., & Torras, C. (2000). An efficient algorithm for searching implicit and/or graphs with cycles. *Artificial Intelligence, 124*(1), 1–30.

Jones, R. E., & Wilson, R. H. (1996) A survey of constraints in automated assembly planning. In: *Proceedings of the IEEE international conference on robotics and automation* (pp. 1525–1532) Minneaopolis (MN), USA.

Kim, J. Y., & Cho, H. S. (2000). A neural net-based assembly algorithm for flexible parts assembly. *Journal of Intelligent and Robotic Systems, 29*(2), 133–160. doi:10.1023/A:1008115522778.

Kuniyoshi, Y., Inaba, M., & Inoue, H. (1994) Learning by watching: Extracting reusable task knowledge from visual observation of human performance. *IEEE transactions on robotics and automation, 10*(6), 799–822. citeseer.ist.psu.edu/kuniyoshi94learning.html.

Lai, H. Y., & Huang, C. T. (2004). A systematic approach for automatic assembly sequence plan generation. *The International Journal of Advanced Manufacturing Technology, 24*(9–10), 752–763. doi:10.1007/s00170-003-1760-5.

Latombe, J. C. (1991). *Robot motion planning, vol. SECS 0124*. Dordrecht, The Netherlands: Kluwer.

LaValle, S. M. (2006). *Planning algorithms*. Cambridge University Press, Cambridge, UK. Available at http://planning.cs.uiuc.edu/.

Le, D. T., Cortés, J., Siméon, T. (2009). A path planning approach to (dis)assembly sequencing. In: *Proceedings of the fifth IEEE international conference on automation science and engineering* (pp. 286–291).

Lee, B., & Saitou, K. (2003). Decomposition-based assembly synthesis for in-process dimensional adjustability. *Journal of Mechanical Design, 125*(3), 464–473. doi:10.1115/1.1587746.

Lee, S., & Moradi, H. (1999) Disassembly sequencing and assembly sequence verification using force flow networks. In: *Proceedings of the IEEE international conference on robotics and automation* (ICRA) (pp. 2762–2767).

Lee, S., & Shin, Y. G. (1990). Assembly planning based on geometric reasoning. *Computation and Graphics, 14*(2), 237–250.

Lefebvre, T., Bruyninckx, H., & Schutter, J. D. (2003). Active sensing for the identification of geometrical parameters during autonomous compliant motion. In: *Proceedings IEEE international conference on robotics and automation* (ICRA) (pp. 2599–2604).

Lefebvre, T., Bruyninckx, H., & Schutter, J. D. (2005). Task planning with active sensing for autonomous compliant motion. *International Journal of Robotic Research, 24*(1), 61–81.

Lefebvre, T., Xiao, J., Bruyninckx, H., & de Gersem, G. (2005). Active compliant motion: A survey. *Advanced Robotics, 19*(5), 479–499.

Lu, T., Zhang, B., & Jia, P. (1993). Assembly sequence planning based on graph reduction. In: *Proceedings of the IEEE TENCON* (pp. 119–122)

Mahanti, A, & Bagchi, A (1985). And/or graph heuristic search methods. *Journal of Association for Computing Machinery, 32*(1), 28–51.

Mantripragada, R., & Whitney, D. E. (1998). The datum flow chain: A systematic approach to assembly design and modeling. *Research in Engineering Design, 10*(3), 150–165. doi:10.1007/BF01607157.

Marian, R. M. (2003) *Optimisation of assembly sequences using genetic algorithms*. PhD thesis, University of South Australia, Adelaide, Australia

Marian, R. M., Luong, L. H. S., & Abhary, K. (2006). A genetic algorithm for the optimisation of assembly sequences. *Computers & Industrial Engineering, 50*, 503–527. doi:10.1016/j.cie.2005.07.007.

Martelli, A., & Montanari, U. (1973). Additive and/or graphs. In: *Proceedings of the third international joint conference on artificial intelligence* (pp. 1–11).

Martelli, A., & Montanari, U. (1978). Optimizing decision trees through heuristically guided search. *Communications of the ACM, 21*(12), 1025–1039.

Meeussen, W., Xiao, J., Schutter, J. D., Bruyninckx, H., & Staffetti, E. (2004). Automatic verification of contact states taking into account manipulator constraints. In: *Proceedings of the IEEE international conference on robotics and automation* (ICRA) (pp. 3583–3588). doi:10.1109/ROBOT.2004.1308808.

Meeussen, W., Schutter, J. D., Bruyninckx, H., Xiao, J., & Staffetti, E. (2005). Integration of planning and execution in force controlled compliant motion. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems* (IROS) (pp. 1217–1222). doi:10.1109/IROS.2005.1545360.

Meeussen, W., Rutgeerts, J., Gadeyne, K., Bruyninckx, H., & Schutter, J. D. (2007). Contact-state segmentation using particle filters for programming by human demonstration in compliant-motion tasks. *IEEE Transactions on Robotics, 23*(2), 218–231.

Mihaylova, L., Lefebvre, T., Staffetti, E., Bruyninckx, H., & Schutter, J. D. (2001). Tracking contact transitions during force-controlled compliant motion using an interacting multiple model estimator. In: *Proceedings of the IEEE international conference on advanced robotics* (pp. 665–670).

Möhring, R. H., Skutella, M., & Stork, F. (2004). Scheduling with and/or precedence constraints. *SIAM Journal on Computing, 33*(2), 393–415.

Mosemann, H., Bierwirth, T., Wahl, F. M., & Stoeter, S. (2000). Generating polyhedral convex cones from contact graphs for the identification of assembly process states. In: *Proceedings of the IEEE international conference on robotics and automation* (ICRA) (pp. 744–749).

Motavalli, S., & Islam, A. U. (1997). Multi-criteria assembly sequencing. *Computer and Industrial Engineering, 32*(4), 743–751. doi:10.1016/S0360-8352(97)00014-4.

Naphade, K. S., Storer, R. H., & Wu, S. D. (1999a). *Graph-theoretic generation of assembly plans, part i: Correct generation of precedence graphs*. Technical Report IMSE Technical Report 99T-003, Lehigh University, Bethlehem, Pennsylvania.

Naphade, K. S., Storer, R. H., & Wu, S. D. (1999b). *Graph-theoretic generation of assembly plans, part ii: Problem decomposition and optimization algorithms*. Technical Report IMSE Technical Report 99T-003, Lehigh University, Bethlehem, Pennsylvania.

Nilsson, N. J. (1980). *Principles of Artificial Intelligence*. Palo Alto: Tioga.

Raghavan, V., Moloineros, J., & Sharma, R. (1999). Interactive evaluation of assembly sequences using augmented reality. *IEEE Transactions on Robotics and Automation, 15*(3), 435–449.

Ramos, C., ao Rocha, J., & Vale, Z. (2001). A complete complexity study of one-processor assembly and manufacturing planning tasks. In: *Proceedings of the 4th IEEE international symposium on assembly and task planning* (pp. 369–374).

Remde, A., Henrich, D., & Wörn, H. (1999). Manipulating deformable linear objects—contact state transitions and transition conditions. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems* (vol. 3, pp. 1450–1455).

Remde, A., Henrich, D., & Wörn, H. (2000) Manipulating deformable linear objects—Force based detection of contact state transitions. In: *Proceedings of the IEEE international conference on robotics and automation* (ICRA) (pp. 1480–1486).

Romney, B. (1997). Atlas: An automatic assembly sequencing and fixturing system. In: *Proceedings of the international conference on theory and practice of geometric modeling* (pp. 397–415).

Romney, B., Godard, C., Goldwasser, M., & Ramkumar, G. (1995). An efficient system for geometric assembly sequence generation and evaluation. In: *Proceedings of the ASME international computers in engineering conference* (pp. 699–712).

Rosell, J. (2004). Assembly and task planning using Petri Nets: A survey. *Proceedings of the I MECH E Part B journal of engineering manufacture*, (vol. 218, pp. 987–994). doi:10.1243/0954405041486019.

Schwarzer, F., Schweikard, A., & Joskowicz, L. (2000). Efficient linear unboundedness testing: Algorithm and applications to translational assembly planning. *The International Journal of Robotics Research, 19*, 817–834. doi:10.1177/02783640022067193.

Schweikard, A., & Schwarzer, F. (1998). Detecting geometric infeasibility. *Artificial Intelligence, 105*(1-2), 139–159.

Sebaaly, M., Fujimoto, H., & Mrad, F. (1996). Linear and non-linear assembly planning: fuzzy graph representation and GA search. In: *Proceedings of the 1996 IEEE international conference on robotics and automation* (pp. 1533–1538).

Smith, S. S. F. (2004). Using multiple genetic operators to reduce premature convergence in genetic assembly planning. *Computers in Industry, 54*(1), 34–49. doi:10.1016/j.compind.2003.08.00.

Staffetti, E., Ros, L., & Thomas, F. (1999a). Finding infinitesimal motions of objects in assemblies using grassmann-cayley algebra. In: *Proceedings of the tenth world congress on the theory of machine and mechanisms* (pp. 584–591).

Staffetti, E., Ros, L., & Thomas, F. (1999b). A simple characterization of the infinitesimal motions separating general polyhedra in contact. In: *Proceedings of the 1999 IEEE international conference on robotics and automation* (ICRA) (pp. 571–577).

Staffetti, E., Meeussen, W., & Xiao, J. (2005). A new formalism to characterize contact states involving articulated polyhedral objects. In: *Proceedings of the 2005 IEEE international conference on robotics and automation* (ICRA) (pp. 3619–3626).

Sundaram, S., Remmler, I., & Amato, N. (2001). Disassembly sequencing using a motion planning approach. In: *Proceedings of the IEEE international conference on robotics and automation* (ICRA) (pp. 1475–1480).

Suzuki, T., Kanehara, T., Inaba, A., & Okuma, S. (1993). On algebraic and graph structural properties of assembly petri net. In: *Proceedings of the IEEE international conference on robotics and automation* (ICRA) (pp. 507–514).

Thomas, F., & Torras, C. (1992). Inferring feasible assemblies from spatial constraints. *IEEE Transactions on Robotics and Automation, 8*(2), 228–239.

Thomas, U., Barrenscheen, M., & Wahl, F. (2003). Efficient assembly sequence planning using stereographical projections of c-space

obstacles. In: *Proceedings of the 2003 IEEE international symposium on assembly and task planning (ISATP 2003)* (pp. 96–102).
Tseng, H. E., & Li, R. K. (1999). A novel means of generating assembly sequences using the connector concept. *Journal of Intelligent Manufacturing, 10*(5), 423–435. doi:10.1023/A:1008971030395.
Tseng, H. E., Li, J. D., & Chang, Y. H. (2004). Connector-based approach to assembly planning using a genetic algorithm. *International Journal of Production Research, 42*, 2243–2261.
Wang, J., Liu, J., & Zhong, Y. (2005). A novel ant colony algorithm for assembly sequence planning. *The International Journal of Advanced Manufacturing Technology, 25*(11–12), 1137–1143. doi:10.1007/s00170-003-1952-z.
Wang, L., Keshavarzmaneh, S., Feng, H. Y., & Buchal, R. O. (2008). Assembly process planning and its future in collaborative manufacturing: a review. *The International Journal of Advanced Manufacturing Technology, 41*(1–2), 132–144. doi:10.1007/s00170-008-1458-9.
Wang, W. P., & Tseng, H. E. (2009). Complexity estimation for genetic assembly sequence planning. *Journal of the Chinese Institute of Industrial Engineers, 26*(1), 44–52.
Wilson, R., & Rit, J. (1990) Maintaining geometric dependencies in an assembly planner. In: *Proceedings of the 1990 IEEE international conference on robotics and automation* (vol. 2, pp. 890–895) Scottsdale, AZ.
Wilson, R. H., & Latombe, J. C. (1994). Geometric reasoning about mechanical assembly. *Artificial Intelligence, 71*(2), 371–396.
Wilson, R. H., Kavraki, L., Lozano-Pérez, T., & Latombe, J. C. (1995). Two-handed assembly sequencing. *International Journal of Robotics Research, 14*(4), 335–350.
Wolter, J. (1991). A combinatorial analysis of enumerative data structures for assembly planning. In: *Proceedings of the IEEE international conference on robotics and automation* (pp 611–618). citeseer.ist.psu.edu/wolter91combinatorial.html
Wolter, J., Chakrabarty, S., & Tsao, J. (1991). *Mating constraint languages for assembly sequence planning*. Technical Report 91-038, Texas A&M University.
Wolter, J. D. (1989). On the automatic generation of assembly plans. In: *Proceedings of the 1989 IEEE international conference on robotics and automation* (vol. 1, pp. 62–68) Scottsdale, AZ.
Xiao, J. (1993). Automatic determination of topological contacts in the presence of sensing uncertainties. In: *Proceedings of the IEEE international conference on robotics and automation* (ICRA) (pp. 65–70).
Xiao, J., & Ji, X. (2000). A divide-and-merge approach to automatic generation of contact states and planning of contact motion. In: *Proceedings of the IEEE international conference on robotics and automation* (ICRA) (pp. 750–756).
Xiao, J., & Ji, X. (2001). Automatic generation of high-level contact state space. *The International Journal of Robotics Research, 20*(7), 584–606.
Xiao, J., & Liu, L. (1998). Contact states: Representation and recognizability in the presence of uncertainties. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems* (pp. 1151–1156).
Yuan, X. (2002). An interactive approach of assembly planning. *IEEE Transactions on Systems, Man, and Cybernetics Part A Systems and Humans, 32*(4), 522–526.
Zha, X. F., Lim, S. Y. E., & Fok, S. C. (1998a). Integrated intelligent design and assembly planning: A survey. *The International Journal of Advanced Manufacturing Technology, 14*(9), 664–685. doi:10.1007/BF01192287.
Zha, X. F., Lim, S. Y. E., & Fok, S. C. (1998b). Integrated knowledge-based petri net intelligent flexible assembly planning. *Journal of Intelligent Manufacturing, 9*(3), 235–250. doi:10.1023/A:1008862631701.
Zheng, Y., Pei, R., & Chen, C. (1991). Strategies for automatic assembly of deformable objects. In: *Proceedings of 1991 IEEE international conference on robotics and automation* (vol. 3, pp. 2598–2603). doi:10.1109/ROBOT.1991.132019.