Computer-Aided Design 67–68 (2015) 58–86

Contents lists available at ScienceDirect

Computer-Aided Design

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

Review

Review and taxonomies of assembly and disassembly path planning problems and approaches☆

Somayé Ghandi, Ellips Masehian*

Faculty of Engineering, Tarbiat Modares University, Tehran, 14115-143, Iran

HIGHLIGHTS

- State-of-the-art review of the Assembly/Disassembly Path Planning (APP/DAPP) field.
- New taxonomies for categorizing APP/DAPP problem types and solution methods.
- Critical discussions on research trends, applications and open problems in APP/DAPP.

ARTICLE INFO

Article history:
Received 22 June 2014
Accepted 1 May 2015

Keywords:
Assembly planning
Assembly path planning
Disassembly path planning
Taxonomy

ABSTRACT

Assembly Planning (AP) is one of the most important elements of process planning in manufacturing industries, and is defined as the process of creating a detailed assembly plan to craft a whole product from separate parts considering the final product geometry, available resources, fixture design, feeder and tool descriptions, etc. AP has three main subproblems: (1) Assembly Sequence Planning (ASP), in which a sequence of collision-free operations is computed for bringing assembly parts together, (2) Assembly Line Balancing (ALB), in which some groups of subassemblies are formed and assigned to assembly stations in a way that their workloads are balanced, and (3) Assembly Path Planning (APP), in which collision-free paths for adding parts to a subassembly are computed. Each of the above subproblems has a disassembly version, creating DASP, DALB, and DAPP problems. All of the above problems have proven to be either NP-hard or NP-Complete, and many researches have been conducted to solve them efficiently. While some surveys and reviews exist on the ASP/DASP and ALB/DALB problems, no comprehensive survey exists for APP/DAPP problems, despite their important role in the design process of products as invaluable tools for deploying concurrent engineering, end-of-life processing, maintenance and repair, and decreasing the cost and time of manufacturing products. This paper investigates the relations between the above six subproblems and reviews the state-of-the-art of the APP and DAPP problems and their solution approaches. Through two new taxonomies the properties and categories of APP/DAPP problems and solution approaches are identified and described, the characteristics and applications of the reviewed 60 most relevant works are exposed and analyzed comprehensively, and open problems in the field are identified.

Contents

☆ This paper has been recommended for acceptance by Ming C. Lin.

* Corresponding author.
E-mail addresses: s.ghandibidgoli@modares.ac.ir (S. Ghandi), masehian@modares.ac.ir (E. Masehian).

http://dx.doi.org/10.1016/j.cad.2015.05.001
0010-4485/

1. Introduction

Assembly Planning (AP) is the process of creating a detailed assembly plan to craft a whole product from separate parts by taking into account the final product geometry, available resources to manufacture that product, fixture design, feeder and tool descriptions, etc. Assembly planning is one of the most important processes in manufacturing products since assembly processes use up to 50% of the total production time and more than 20% of the total manufacturing cost [1]. So, efficient assembly plans can reduce manufacturing time and costs significantly. The Assembly Planning problem has been shown to be an NP-complete problem [2] and covers three main subproblems: Assembly Sequence Planning, Assembly Line Balancing, and Assembly Path Planning.

The *Assembly Sequence Planning* (ASP) problem concerns with finding a sequence of collision-free operations $o_1, \ldots, o_n$ that bring the assembly parts $p_1, \ldots, p_n$ together, having given the geometry of the final product A and the positions of parts in the final product. A systematic overview on the ASP is presented in [3], which includes a survey of the elements of sequence planning, such as finding a feasible sequence, determining an optimal sequence according to one or more operational criteria, representing the space of feasible assembly sequences in different ways, applying search and optimization algorithms, and satisfying precedence constraints existing between subassemblies. The ASP problem is classified as an NP-hard problem and so cannot be solved in polynomial time [4]. The complexity of finding an optimal sequence increases linearly with the size of the space of all potential assembly sequences in the case of exhaustive search. However, when all created sequences with the assembly sequence planner are linear and monotone, the number of assembly operations equals the number of parts, n, and therefore the total number of potential sequences is given by the permutations of parts, $n!$. Assuming that all the sequences created by the assembly sequence planner are monotone, the size of the solution space amounts to $(2n-2)!/(n-1)!$, and if non-monotone sequences are considered as well, the number of potential sequences will be infinite [5].

The *Assembly Line Balancing* (ALB) problem deals with partitioning the total assembly operations into a set of n elementary tasks o_i $(i = 1, \ldots, n)$ with times t_i, and assigning them to m assembly workstations w_k $(k = 1, \ldots, m)$ such that in all workstations approximately equal assembly times are spent and the precedence constraints between operations are satisfied. Assuming that the set S_k of tasks is assigned to the workstation k, the assembly time of that workstation equals $t(S_k) = \sum_{j \in S_k} t_j$. The ALB problem is also NP-hard [6], and can be divided into two categories: Simple Assembly Line Balancing Problem (SALBP), and Generalized Assembly Line Balancing Problem (GALBP). SALBP is appropriate for modeling assembly lines with all their input parameters deterministically known, which produce a unique model of a single product (i.e., serial assembly lines) [7]. On the other hand, GALBP is appropriate for balancing more complex lines such as parallel, U-shaped, mixed-shaped, and two-sided lines with stochastic dependent processing times [8]. A survey on researches in ASP and SALBP that have applied soft computing approaches is presented in [9], covering the years 2001 to 2011. Soft computing approaches are useful for ASP and ALB optimization because they are able to handle more complex and large size problems with numerous constraints.

The *Assembly Path Planning* (APP) problem considers generating sequences of positions for parts $p_1, \ldots, p_n$ of a final product A with known geometries, from an initial (disassembled) position to the final (assembled) position, in the form of paths $\tau_1, \ldots, \tau_n$ that contain no collisions between assembled and disassembled parts and with obstacles $o_1, \ldots, o_r$ in a workspace $W \in \Re^2$ or $\Re^3$. The

APP problem inherently resembles the classic "Piano Movers' Problem" in robot motion planning, which is simply described as moving a solid object in 3D space with six degrees of freedom amongst stationary obstacles, and has been proved to be PSPACE-hard and NP-Complete [10–12]. Therefore, APP has the same complexity of the general motion planning. Assembly paths are calculated based on the assembly sequences that are the output of the ASP problem. APP is an important phase of the broader Assembly Planning problem, because: (1) the solution of this problem can provide better feedback by CAD systems to designers [13], (2) a good assembly path plan would ensure production of products that are more cost effective to manufacture and thus can influence the efficiency of the overall assembly planning process, and (3) a poor assembly path plan can increase the cost of manufacturing significantly and reduce the productivity since long part travel paths consume more time and energy. APP can be formulated as a motion planning problem [14,15] using the notion of *Configuration Space* (C-space) proposed by Lozano-Pérez in 1983 [16]. A configuration q is a minimal set of parameters defining the location of a mobile system in the world, and the configuration space C is the set of all configurations. In the case of a system M involving n mobile objects m_i (e.g. the parts of the assembly), the *Composite Configuration Space* C is the Cartesian product of the configuration spaces of all the objects. That is, $C = \prod_{i=1}^{n} c_{m_i}$; $i = 1, \ldots, n$. Given the initial configuration q_{dis}, the problem consists in finding a feasible path in C from q_{dis} to a final assembled configuration q_{ass}.

Another closely related issue to the Assembly Planning is *Disassembly Planning* (DAP). Given the geometry of the final product and the resources to manufacture that product, Disassembly Planning creates a manufacturing plan for removing specific parts from a whole assembly. 'Assembly by Disassembly' is an important strategy for assembling since parts in an assembled state of a product have far more precedence and motion constraints than in a disassembled state of that product [17]. These constraints drastically reduce the solution space size of the ASP and APP problems. According to this strategy, an assembly plan is obtained by disassembling a whole product into its constituting parts and then reversing the order of disassembly. When only geometric constraints are concerned and all parts are rigid, there is a bijection between assembly and disassembly sequences and paths, though this bijection does not remain correct when physics (e.g. gravity, friction) and motion control uncertainty are taken into account, or when some parts are deformable [18] or toleranced [19]. For example, disassembly of a part assembled by rivets is not simply the reverse of its assembly. DAP concerns with feasibly removing specific parts from a whole assembly, and is one of the important processes in product manufacturing since it is required for maintenance and repair of a product, is used in reverse and concurrent engineering, and has a major role in devising a proper design process.

A classification of DAP methods for disassembling of a component C_i from a product A is presented in [20], according to the following criteria: (1) *1- vs. m-disassembly*: C_i is m-disassemblable if multi-step translational motions are needed to remove it from A; (2) *Direct vs. Indirect disassembly*: C_i is directly disassemblable if it can be removed from A without removing other components; (3) *Sequential vs. Parallel (non-sequential) disassembly*: sequentiality refers to the maximum number of moving subassemblies with respect to one another in a disassembly method; (4) *Monotonic vs. Non-monotonic disassembly*: in monotonic disassembly the components are totally removed from A, and conversely, non-monotonic disassembly requires partial disassembly of one or more components; (5) *Complete vs. Selective disassembly*: complete disassembly occurs when all parts of the product A are disassembled; (6) *Destructive vs. Non-destructive disassembly*: the disassembly method is destructive when one or more components must be destroyed during the operation.

DAP covers three main subproblems: Disassembly Sequence Planning, Disassembly Line Balancing, and Disassembly Path Planning, all of which are either NP-hard or NP-complete:

The *Disassembly Sequence Planning* (DASP) problem computes a sequence of collision-free operations that remove the assembly parts from the final product, having given the geometry of the final product and the positions of parts in the final product. A survey on this subproblem is presented in [21], which covers topics such as different ways of representing the space of feasible disassembly sequences; component-oriented approaches that consider automatic path generation, motion and stability analyses and collision detection; product-oriented approaches that perform automatic analyses of the ability to decompose a product from its assembly drawing; and the hierarchical-tree approach which is based on the inverse Material Requirement Planning (MRP) and relates disassembly processes to the hierarchical product structure.

The *Disassembly Line Balancing* (DALB) problem optimally assigns the required disassembly operations of a product to disassembly stations so that the precedence constraints between operations are satisfied. DALB is necessary when some returned products are disassembled in order to recycle and remanufacture that product. Many papers on this topic have been published to date, for example, Scholl has presented a comprehensive textbook on this problem [22].

The *Disassembly Path Planning* (DAPP) problem computes the path for separating a part from a sub-assembly, having given the geometry of the final product, the positions of parts in the final product, and the space in which the assembly operation is performed. Given the initial configuration q_{ass}, the DAPP tries to find a feasible path in the composite configuration space C from q_{ass} to a final disassembled configuration q_{dis}. However, the difference between the DAPP and standard path planning problems is that the final disassembled configuration may not be precisely specified, but implicitly defined by the distances between parts. Like APP, DAPP is an important and useful tool for designing assembly/disassembly processes and reducing manufacturing time and cost.

APP and DAPP problems are tightly related to four more specific and limited problems, which can be solved if solutions to the APP and DAPP are found. These problems are Motion Stability, Assembly Maintainability, Selective Disassembly Planning, and Partitioning. Among these problems, only the Motion Stability Problem is related to both APP and DAPP problems, and the others are closely related only to the DAPP problem.

(1) *Motion stability problem* deals with identifying the presence and time of occurrence of instability of an assembly during (dis)assembly part motions. The first algorithm presented for analyzing the stability of a number of blocks and solving the set of contact forces in the assembly of rigid blocks was presented in 1970 by Blum et al. [23]. Boneschanscher et al. considered the situation in which physical and insertion forces are present in the assembly process and developed a semi-heuristic algorithm to determine the stability of sub-assemblies [24]. An investigation of the computational complexity of stability of polygons was presented in [25]. Also, optimum locations of fixels (fixturing points) for assemblies were determined by the potential energy minimization principle in [26]. For identifying the set of orientations under which an assembly is stable when gravity is considered, Mattikalli et al. [27,28] developed linear programming approaches. Mosemann et al. first created an AND/OR assembly graph in [29] and then analyzed the stability of the subassemblies at each node of the AND/OR assembly graph in [30]. Recently, Rakshit and Akella have presented an approach for the motion stability analysis of mechanical parts disassembly in the presence of physical forces such as gravity and friction [18].

(2) *Assembly maintainability problem* investigates the possibility of removing a particular part from an assembly without chang-

ing positions and orientations of other parts, and finds such a path if one exists. Before 1995, Assembly Maintainability was labor intensive and heavily relied on humans in providing access paths for parts through using either physical mock-ups or computer animations with CAD models. However, Chang and Li presented an automated approach in 1995 to replace this manual process and demonstrated the feasibility of using an automated assembly maintainability system [31]. Although both the Assembly Maintainability and DAPP problems have the same goal of finding paths for disassembling a product, they differ in that DAPP focuses on finding disassembly paths only at certain straight directions, but the Assembly Maintainability problem accommodates complex straight or curved part removal motions. The Assembly Maintainability problem can be reduced to the "piano movers" problem, and since it is NP-complete the Assembly Maintainability problem is also NP-complete.

(3) *Selective disassembly planning problem* tries to generate a sequence of parts removal required to disassemble a certain 'target' part from an assembly. This operation is necessary when the target part has reached its expected end of life or because it fails to function properly in the simulation of maintenance operations. Two subproblems must be solved to generate a selective disassembly plan: the DASP and DAPP problems [32]. Because these two subproblems are NP-complete, the general Selective Disassembly Planning problem is also NP-complete.

(4) *Partitioning problem* concerns with finding a proper subset S of a rigid parts assembly A and a direction d such that the complementing subsets S and $A\backslash S$ do not collide with each other when the subassembly S is moved as a rigid body along d. The partitioning problem is NP-complete for two-handed monotone assembly plans (i.e. when the given assembly can be partitioned into two complementing subsets each treated as a rigid body without intermediate placement of subassemblies) and arbitrary direction [33,34].

Fig. 1 illustrates the relations between the main six problems of ASP, ALB, APP, DASP, DALB, and DAPP, together with their short descriptions and the objective functions that have been considered and optimized in the literature for each problem. Note the logical sequence of ASP→ALB→APP and DASP→DALB→DAPP shown in the figure, as the output of a problem can be the input information for its succeeding problem.

1.1. The taxonomies

Surprisingly, while some surveys and reviews exist on the ASP/DASP and ALB/DALB problems, the APP/DAPP problems or solution methods have not received due attention despite their important role in the design process of products as invaluable tools for deploying concurrent engineering, end-of-life processing, maintenance, repair, and decreasing the cost and time of manufacturing products. Therefore, regarding the importance of the APP/DAPP problems and the vast body of research conducted on these topics on one hand, and the lack of a consolidated and unifying framework or a proper categorization of the problems and solution approaches of the APP/DAPP problems on the other hand, developing a comprehensive survey and classification seems indispensable.

Trying to address this issue, in this paper we present a state-of-the-art review on the APP and DAPP problems and their solution approaches. Also, through two new detailed taxonomies, properties and categories of APP/DAPP problems and solution methods are identified and described, and their characteristics and applications in the reviewed literature are exposed broadly. The proposed taxonomies cover the main aspects of APP/DAPP problems and solutions, as described below:

APP/DAPP problems taxonomy: The first step in defining an APP/DAPP problem is to obtain information on the structure of the assembly and its parts, and then construct a comprehensive model for establishing assumptions and expressing the parts, the assembly, and the relationship between parts in the assembly. Once a model has been built, the nature of the assembly can be identified and analyzed. Therefore, the taxonomy of APP/DAPP problems has two main aspects:

- *Problem model*, which includes assumptions about the considered components in the assembly, dimensions, tolerances, geometry, rigidity, and various geometrical, physical and mechanical constraints of the assembly and its parts, as well as the allowed movements during the assembly/disassembly operation, and the objective function of the model. These features are further described in Section 2.1.
- *Problem nature*, which expresses the inherent features of the (dis)assembly problem stemmed from the geometrical structure of the assembly, such as number of parts, sequentiality, monotonicity, linearity, coherence and scale. These features are further described in Section 2.2.

APP/DAPP solution methods taxonomy: Given a sequence for assembling/disassembling parts to/from an assembly (obtained by solving an ASP/DASP problem), a solution to the APP/DAPP problem is a set of plans for the motions of some or all parts of the assembly from their initial to final configurations. This can be done using several path/motion planning methods which are selected according to their characteristics, nature, and ability in satisfying the assumptions and constraints of the problem model. APP/DAPP solution methods can be studied from two aspects:

- *Solution approach*, which is the course of action the planner adopts for finding (dis)assembly paths and coming up with a total (dis)assembly plan that solves the APP/DAPP problem. The approach could be Graph-based, Grid-based, Sampling-based, Space Decomposition, or Interactive, as described in Section 3.1.
- *Solution nature*, which refers to inherent properties of the solution methods, such as mode, scope and completeness. Section 3.2 presents detailed explanations about these features.

The key benefits of these taxonomies are:

1. They are efficient and consistent representations of the sizeable volume of research and information in the APP/DAPP field.
2. They incorporate and provide essential keywords and their hierarchical structures in all important concepts within the APP/DAPP field.
3. They highlight the main features of many researches in the literature and attribute various characteristics to them though not directly and explicitly mentioned in the original articles.
4. They provide sufficient knowledge and guidance for understanding and directly finding correct and exact information about researches on each aspect of APP/DAPP problems and solution methods.
5. Through using comprehensive frameworks of the taxonomies and the discussions provided, readers can identify research gaps in APP/DAPP problems formulation and solution and thereby initiate novel researches in this field.

In this paper, along with describing the above two taxonomies, the APP/DAPP literature has been critically reviewed to sufficient details and all industrial or puzzle-like applications mentioned in the reviewed papers are reported in a table, together with a summary of their main features. The survey revealed some remarkable facts about research on APP/DAPP, such as: (1) Assembly of flexible parts has been hardly addressed; (2) Assembling tools, hands, or fixtures have mostly been ignored in planning assem-

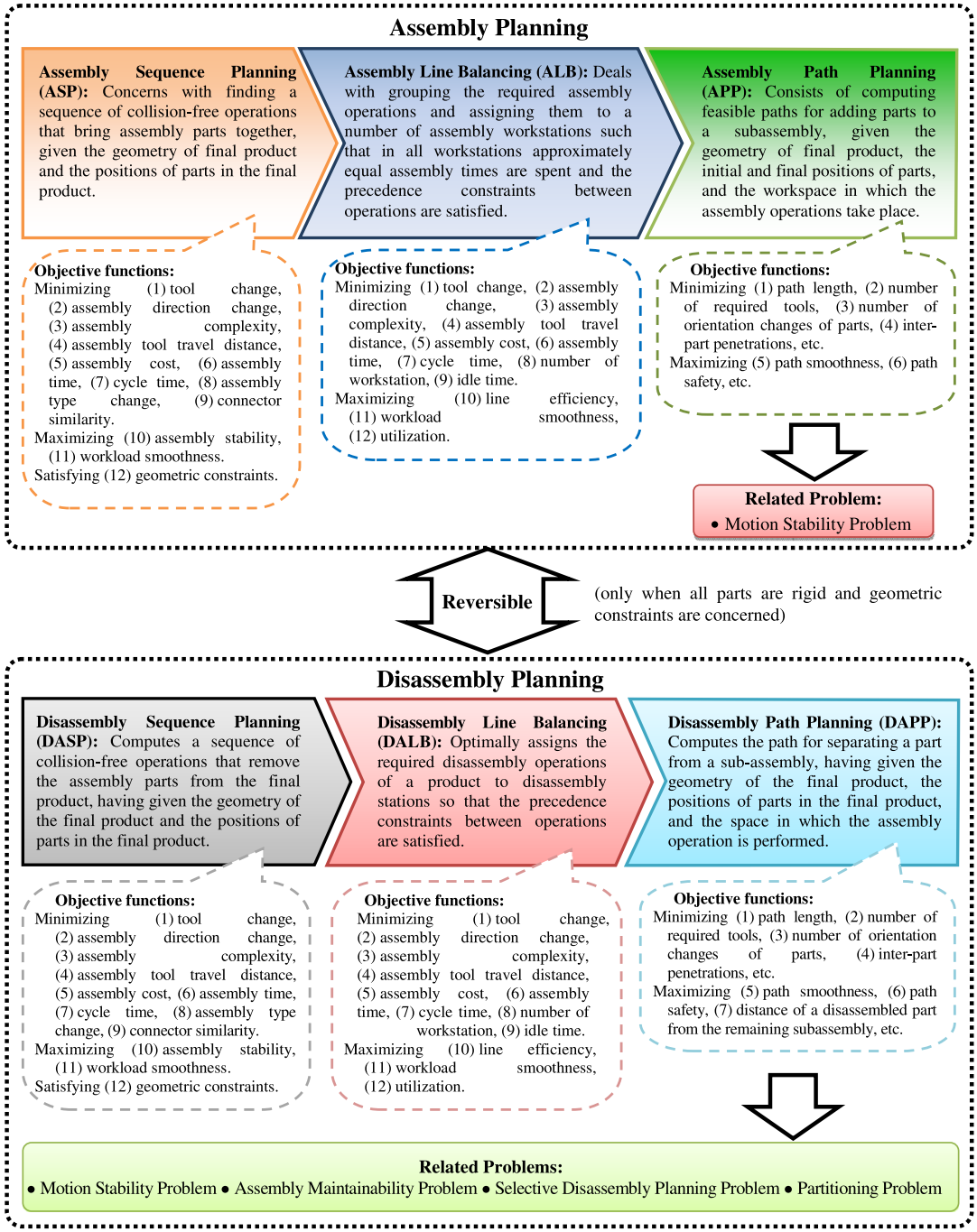

Fig. 1. Assembly and Disassembly Planning subproblems and their relations.

bly sequences and paths; (3) There are few solution methods that can handle non-monotone, nonlinear, non-sequential, and incoherent assembly planning problems; and (4) About 65% of the studied 119 examples were actual or simulated industrial assemblies and the rest were puzzle-like, theoretical problems. Other interesting facts are presented through the discussions presented in Section 4, where some analyses regarding the proportion of works on APP/DAPP solution approaches and their chronological trends, as well as open problems in the field are also presented. Finally, conclusions are provided in Section 5. Additionally, due to the abundance of acronyms coined for various solution methods, we have tabulated all the acronyms used in this article in Appendix.

2. Taxonomy of APP/DAPP problems

In this section we present a taxonomy for the main features of APP/DAPP problems and review the articles that have dealt with these features. Fig. 2 illustrates the taxonomy with its two main aspects of problem model and problem nature.

2.1. Problem model

An APP/DAPP problem concerns with finding a set of plans for the motions of some or all parts of the assembly from an initial configuration to their final configuration. Therefore, the most rudi-

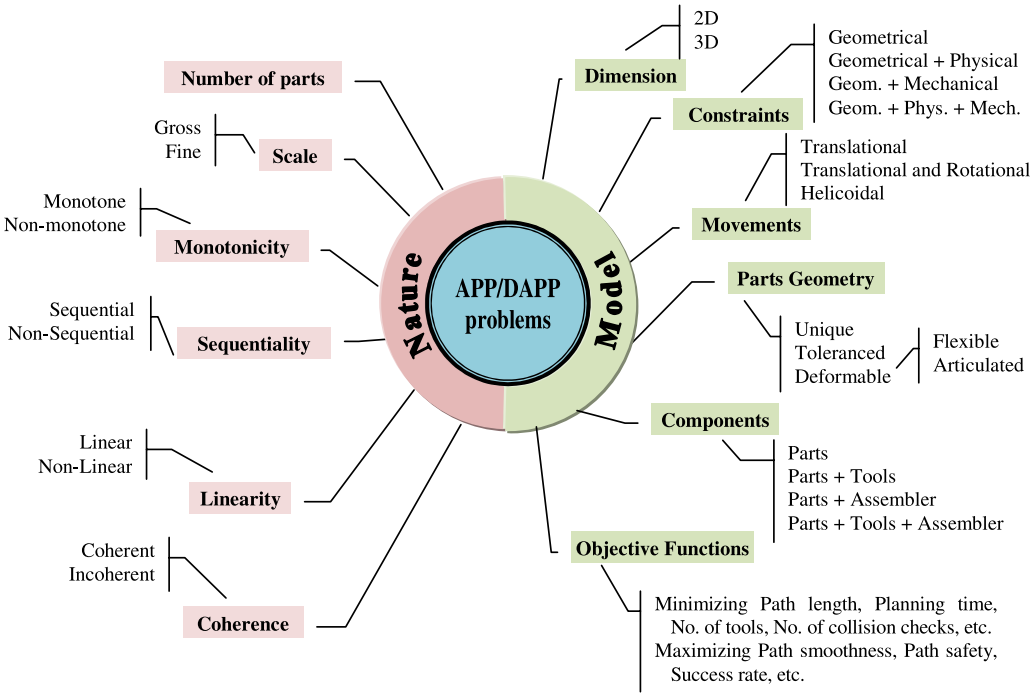

Fig. 2. Taxonomy of APP/DAPP problems.

mentary issue to be considered is the dimension, geometry, and constraints of the parts that constitute the assembly. Other features of an APP/DAPP problem include the limitations of the workspace where the task is performed, and the type of moves the parts are allowed to make. In modeling a problem, some questions must be clearly answered and considered: Are the parts able to 'fly' independently toward/from an assembly, or they should be moved manually or by means of a robotic manipulator? Is there sufficient space for moving/removing parts to/from an assembly freely? Are the parts assumed to be completely rigid, or they may have toleranced or deformable geometry? What are the constraints governing the mutual relationships between interacting parts: only geometrical (preventing intersections), or physical (e.g. friction, forces) and mechanical (e.g. stress, strain) constraints as well? This section details the above issues and the researches which have dealt with them.

2.1.1. Dimension

An assembly/disassembly process may be simulated in a two- or three-dimensional Euclidean workspace, represented by $\Re^2$ or $\Re^3$. Since real-world environments for assembly operations are 3D, only a few researches have modeled their (dis)assembly path planners in 2D [18,35–37]. Nevertheless, when parts are simple 2.5D objects located next to (and not atop of) each other, the problem is better to be modeled in 2D.

2.1.2. Components

A real-world assembly task is typically performed with the aid of human or robotic hands (grippers), various tools and equipment, and appropriate worktables, jigs, and fixtures. However, to consider all the involved components in solving ASP/DASP, ALB/DALB, or APP/DAPP problems is virtually impossible due to very high complexity and NP-hardness of these problems. Therefore, the models constructed for solving such problems must simplify and abstract the real-world by relaxing some constraints, such as types of components considered in the model. Indeed, an APP/DAPP problem model can accommodate only parts, or a combination of parts with tools and/or assembly hands (assemblers). The more components are modeled, the more difficult will be the assembly path planning.

Although in real-world no assembly task can be done without the aid of human or robotic hand, aiming to reduce the complexity of APP/DAPP problems, many researchers consider only assembly parts in their model and assume them as free-flying objects that can move in the space among surrounding obstacles, thus avoiding the complications of manipulating parts by robots within tight spaces around the assembly. Therefore, by ignoring tools, fixtures and robots to (dis)assemble parts, an APP/DAPP problem reduces to the classical 'piano movers' problem which tries to compute collision-free paths for 'robots' (i.e., the moving parts of the assembly) among 'stationary obstacles' (i.e., the rest of the assembly).

Sometimes, depending on the application, considering just the parts is not sufficient. In assembling complex products, assembly resources like fixtures and tools must enter the assembly environment together with parts in order to rapidly finish the task. Moreover, many mechanical assembly operations require the use of various tools to manipulate, attach, and test parts and subassemblies. The great majority of common assembly tools are *post-tools*, which are those applied after that the corresponding parts are mated, such as welders, screwdrivers, wrenches, and most inspection and measurement operations [17]. In modeling an APP/DAPP problem, if the involvement and existence of assembly resources is not considered by the designer, then an infeasible solution may be generated, leading to failure of the assembly. A few works like [38] have taken into account assembly resources like fixtures, jigs, tools, and operations in finding proper paths for the parts by changing the orientations or locations of the resources when no feasible path for parts are found given the current arrangement of tools and fixtures.

In order to plan even more accurate (dis)assembly paths, one must integrate the assembler (e.g. robot) that manipulates the assembly parts as well. The system composed by the robot grasping a part at a given pose can be regarded as a closed chain mechanism, and possible motions take place in the self-motion manifold of this mechanism [39]. An integrated view of part assembly is presented in [40] where two strategies are introduced to incorporate an

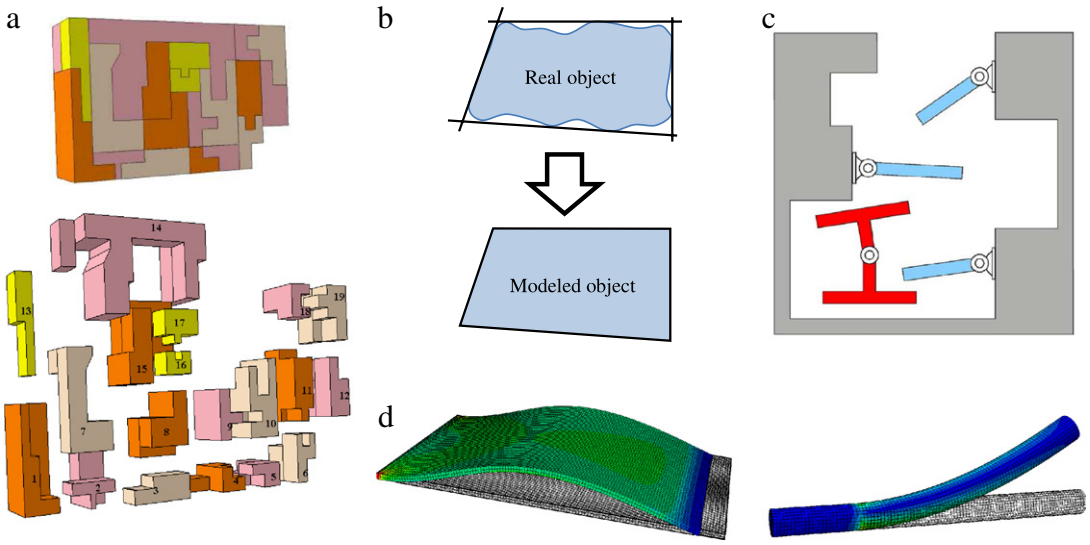

Fig. 3. (a) A complex assembly and exploded view of 19 parts with unique (rigid) geometry; (b) A part with imperfect edges modeled with tolerance geometry (from [19]); (c) Disassembly path planning problem for two objects with articulated parts. The problem consists in finding a path to extract the small (red/dark) object from the big one (from [39]); (d) Parts with flexible geometry, before and after deformation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

articulated assembler: (1) Planning an assembly path for the part alone and then motion planning for the assembler to execute the assembly path via an inverse kinematics operator, (2) (when the first strategy fails) Searching the composite configuration space of both the assembler and the grasped part together, in which case the system will be high-dimensional and the planning task will be more challenging despite such strategy allows dealing with more complicated cases than the previous strategy.

Considering an assembly system comprised of parts, tools, and assemblers simultaneously makes the problem model very complicated, as in the case of real-world mechanical systems. Numerous collisions between interacting physical components with different geometrical specifications are the major source of complexity in such systems. Among our reviewed works, no one addressed this problem explicitly, though [41] considered simple path planning (and not assembly path planning) of a manipulator end-effector in a virtual manufacturing (VM) environment in spot welding of multiple points on fixed metal sheets while avoiding collisions with fixtures, and [42] developed a virtual environment and Virtual Prototyping (VP) method to support disassembly activities performed by a designer involving issues such as representation of the user and his/her movements in the virtual space, as well as representation of the behaviors of tools (such as screwdriver, wrench etc.) and their positioning with respect to the hand.

2.1.3. Part geometry

The geometry of parts of a product could be *Unique*, *Toleranced* or *Deformable*. A part with unique geometry is considered 100% rigid, with no deviation from its mathematical or geometrical model. For example, a long solid rod may be modeled as having a unique geometry, although actually if deflects under its own weight. By considering a relaxation to the perfect rigidity conditions, parts can have Toleranced geometry, which means they are modeled as rigid parts within a controlled predefined imprecision. Inherently imprecise manufacturing processes (like sand casting) produce such parts which may have slightly variable shapes. For example, parts with ignorable details on their surface or negligible dimensional/angular errors are best to be modeled in toleranced geometry. For the first time, Latombe, Wilson, and Cazals considered APP problems for products made of parts with toleranced geometry, and had the important goal of guaranteeing interchangeability of parts in assembly products such that given any set of parts manufactured according to the specified tolerances, a satisfactory assembly would be possible [19]. The following two works deal with toleranced parts, but not in the APP context: in [43] the effect of haptic feedback on a user's ability in detecting motion tolerances in a virtual assembly environment is investigated, and there is an indication of tolerance considerations for assembly parts in [44] which investigated the practical application of the virtual reality (VR) in interactive design and manufacture planning for industrial assembly and disassembly tasks.

Deformable parts can undergo considerable variations in their shape, and may be of two types: *Articulated*, and *Flexible*. Articulated parts are usually composed of one or two links connected to a base by revolute joints (somewhat similar to simple manipulators) and can change shape along certain directions. The only work we found on (dis)assembly path planning of articulated objects is [39], which generalized its solution method to the protein ligand interaction problem. Flexible parts, on the other hand, can freely and reversibly change their shape in the form of tension, compression, twist, or bend in as much as the modulus of elasticity of their substance permits. In many assembly operations the shapes of parts are assumed to be not changing during the process since due to their intrinsic characteristics, flexible parts introduce additional degrees of complexity to the (dis)assembly path planning problem. However, most assembled complex products like ships, aircrafts and automobiles are composed of rigid and flexible parts, and for automatic generation of (dis)assembly plans for such products, the deformability of their flexible parts must be taken into account. (Dis)assembly path planning of flexible objects have been addressed in very few works, including [45] which presented a method for automatically planning a smooth and collision-free path for a wiring harness to be inserted into the engine compartment of a car, and [46] which proposed a method of virtual assembly via haptics to simulate assembly operations of an elastic tube. The method achieved both high-fidelity and high-rendering update rates of simulation by using the physical 'beam-bending' model to simulate the behavior of the elastic tube during assembly and through modeling contact forces based on different types of contact states with the effects of deformation and friction taken into account. Fig. 3 shows parts with unique, toleranced, articulated, and flexible geometries.

Table 1
Relevant works in each feature of Model in APP/DAPP problems.

MODEL	**Dimension**	**2D** [18,35–37,51,60] **3D** [13,17,19,20,31–33,38–42,45,46,48–50,52–59,61–90]
	Components	**Parts** [13,18–20,31–33,35,45,46,48,52,54,57–65,67–71,74–83,86–90] **Parts + Tools** [17,36–38,50,51,53,72,73,85] **Parts + Assembler** [39,40,49,55,56,66,84] **Parts + Tools + Assembler** [41,42]
	Part geometry	**Unique** [13,17,18,20,31–33,35–38,40–42,48–85,87–90] **Toleranced** [19,86] **Articulated** [39] **Flexible** [45,46]
	Movements	**Only translational** [18–20,31,33,35,37,57,61,65,69,73,74,77,79–81,84,87–89] **Translational and rotational** [13,17,32,36,38–42,45,46,48–56,58–60,62–64,66–68,70–72,75,76,78,82,83,85,86,90]
	Constraints	**Geometrical** [13,17,19,20,31–33,35,37–42,45,51–90] **Geometrical + Physical** [23–30,36,45,46,48–50] **Geometrical + Mechanical** **Geometrical + Physical + Mechanical**
	Objective function	**Minimizing path length** [35,51–53] **Maximizing safety of paths** [31,54,55] **Maximizing path smoothness** [45,56] **Minimizing number of orientation changes of parts** [38,51,57] **Minimizing inter-part penetrations** [64] **Maximizing distance of a disassembled part from the remaining subassembly** [40,58,59]

2.1.4. Movements

In many modern manufacturing environments a large part of assembly processes are performed by robots, and regarding the high volume of manipulations performed in an assembly station, any simplification in the trajectory of robots' end-effectors leads to faster and easier task accomplishment, saving substantial resources. In fact, helicoidal movements similar to screw driving and trajectories comprised of translations and rotations are more complex than trajectories which require only translational movements (i.e. parts have fixed orientation), and paths that have minimal changes in direction are preferable, as changing the direction of a subassembly is usually performed very slowly and may require additional expensive facilities. So in most of researches on APP/DAPP, helicoidal movements (as in assembling by screws) are assumed to be merely translational, in order to avoid the complications of this type of movement. However, some (dis)assembly operations are not possible by just translational movements, and a combination of rotation and translation is necessary for moving a part. A few works like [38] have planned proper paths for parts by changing the orientations or locations of the resources (and hence changing the assembly sequence) when no feasible path for the parts could be found given the current arrangement of tools and fixtures.

2.1.5. Constraints

In order for a path planner to automatically generate feasible (dis)assembly plans from just an assembly drawing given by designer, it must consider and intelligently analyze various geometrical, physical, and mechanical constraints regarding the parts and their mutual interactions. Most assembly path planners use only geometric models of parts and assembly, and plan part motions that are merely collision- and intersection-free, whereas some others consider physical properties of the parts and incorporate factors such as friction [18,23,25,27,29,30,36,46–48], gravity [18,23–30], and forces [18,23,25–28,30,36,46,48–50] into the problem model, in addition to the geometrical constraints. Nevertheless, no work has been found to deal with the thermal expansion/contraction physical property, as well as mechanical constraints (such as part deformations under tensional, torsional or compressional stresses) in (dis)assembly operations, although sometimes an assembly task is possible only through considering such mechanical behavior of parts. For instance, pressing a peg into an exactly equal-sized hole cannot be planned without considering geometrical, physical, and mechanical constraints altogether.

2.1.6. Objective function

In solving an APP/DAPP problem the planner seeks to optimize one or more objective functions from different perspectives. Most of the considered objective functions address general aspects of path planning, such as minimizing path length [35,51–53], maximizing path safety and smoothness, etc. Sometimes there are preferred neighborhoods in (dis)assembly environments through which an access path should be found. Access is often limited by realistic concerns of an assembly, such as the size of a moving object, support regions for tools, heat sources to avoid, or simply access convenience. So, generating biased safe paths can be an objective function in APP/DAPP problems [31,54,55]. Sometimes the generated initial (dis)assembly path is very close to the obstacles and has discontinuous curvature. Accordingly, some smoothing procedures have been employed to increase the clearance and smoothness of the initial path, as in [45,56]. Also, minimizing the number of orientation changes of the assembler in the process of assembly can reduce the complexity of manipulation [38,51,57]. Another objective functions that is specific to the DAPP problem is maximizing the distance of a disassembled part from the remaining subassembly, as mentioned in [40,58,59]. Since in disassembly path planning the final configurations of parts are not important, such an objective function is used to describe approximate final positions of disassembled parts which must be far enough from the main assembly and outside its bounding region defined as a sphere, axis-aligned bounding box (AABB), object-aligned bounding box (OBB), or convex hull of the main assembly.

At the end of this subsection, the articles considering various features of APP/DAPP models are summarized in Table 1.

2.2. Problem nature

Problem Nature expresses the inherent features of (dis)assembly problems resulted from geometrical structure of the assembly. When the parts of an assembly have simple geometries then the complexity of the corresponding APP/DAPP problem is measured in terms of the *number of parts*, n, but when the parts have more complex polygonal or polyhedral shapes, then the total number of their vertices N is the appropriate factor to reflect the complexity of the problem. Nonetheless, these measures (i.e., number of parts and their geometries) alone do not express the difficulty of achieving a valid (dis)assembly path plan. Other involved features in the complexity of APP/DAPP problems are Scale, Sequentiality, Mono-

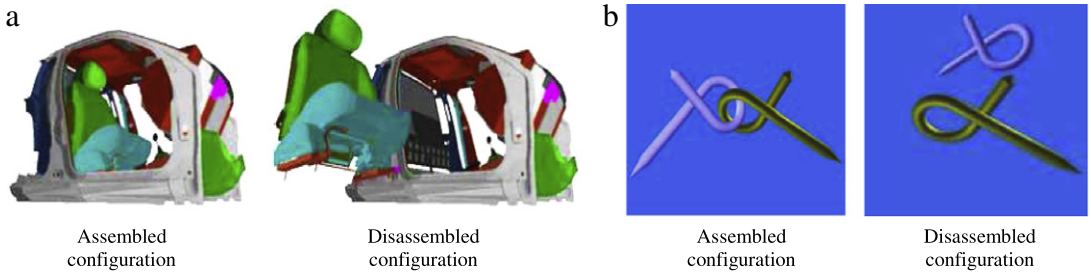

Fig. 4. (a) A Gross disassembly problem (from [40]); (b) A Fine disassembly problem where small positional errors of the parts in the alpha puzzle may cause failure in achieving a collision-free disassembly path for them.
Source: From [13].

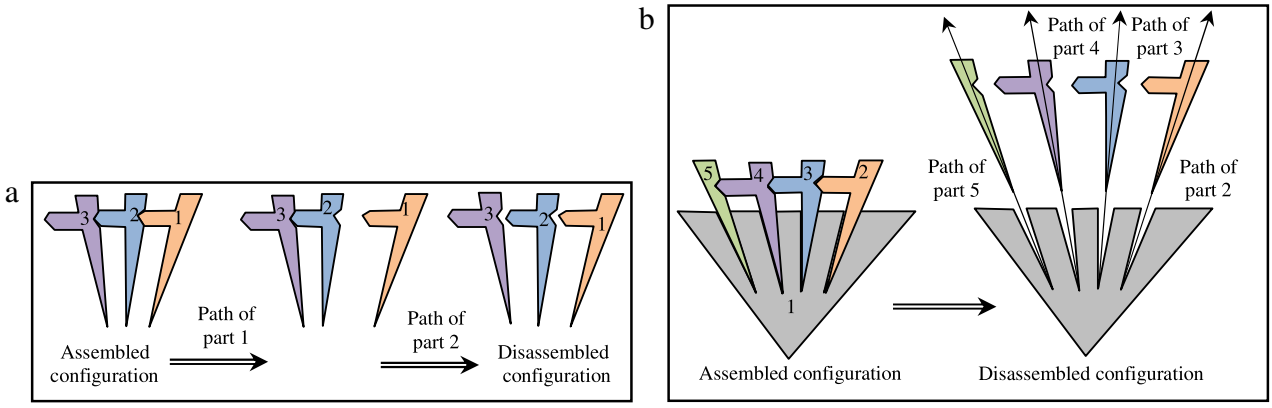

Fig. 5. (a) A sequential, monotone, linear, and contact–coherent disassembly problem; (b) A non-sequential disassembly problem where the number of subassemblies moving with respect to one another in the disassembly operation is five.
Source: From [47].

tonicity, Linearity, and Coherence, as described below. It should be noted that for each complexity feature of an APP/DAPP problem (described in Sections 2.2.2–2.2.5), a corresponding ASP/DASP feature can be defined due to the strong dependence of assembly paths and sequences [79]. That is why the definitions presented in these subsections seem analogous to their corresponding concepts in ASP/DASP problems, which are mentioned in [3].

2.2.1. Scale

The scale of an APP/DAPP problem can be either *Gross* or *Fine*. In a gross APP/DAPP problem the free space between parts of the assembly is much wider than the parts' sizes and so small positional errors in locating parts in the assembly do not lead to missing a feasible path. Carlson et al. presented a gross (dis)assembly path planner by concurrently employing path planning techniques and variation simulation, which was able to compute assembly paths for industrial robots that are less sensitive to geometrical variations existing in the environment and in links and control system of assembly line robots and, through preventing robots' motions in areas with high variation and preferring low-variation zones [56]. In a fine APP/DAPP problem, spaces between parts is so tight that even small positional errors of parts may cause failure in achieving a collision-free (dis)assembly path. Most of (dis)assembly path planners are of this type. Fig. 4 illustrates two examples of APP/DAPP problems with gross and fine scales.

2.2.2. Sequentiality

Sequentiality refers to the maximum number of moving subassemblies with respect to one another in any (dis)assembly operation. The simplest APP/DAPP problems in terms of the number of required hands need two-handed assembly plans, also called *binary* or *sequential* plans. Most researchers have presented sequential (dis)assembly path planners as a result of the difficultly of reasoning about simultaneous relative motions of more than two subassemblies in generating non-sequential (dis)assembly plans, and the fact that most real products can be assembled via sequential (dis)assembly plans, that is, by using just two hands. It is noted that the worktable on which the assembly is placed is considered as one hand.

Fig. 5 illustrates two examples of products with sequential and non-sequential disassembly path plans. Note that the product in Fig. 5(b) cannot be (dis)assembled with sequential (dis)assembly path planning.

Fogel and Halperin presented an exact and efficient algorithm for (dis)assembly of the Split Star Puzzle in 3D which required multiple number of hands [65]. Le et al. described a new method for simultaneous (dis)assembly sequence planning and path planning problems for objects with arbitrary shapes. Although the method was devised for products with sequential (dis)assembly path plans, it could be modified to treat products with non-sequential (dis)assembly path plans [58]. Jin et al. presented an approach for assembly path planning of a decelerator that needed more than two hands to be assembled [83]. Also, Guibas et al. developed an algorithm for solving the polyhedral assembly partitioning problem under infinitesimal motions, when partitioning of parts could not be done with two hands [67].

2.2.3. Monotonicity

Monotonicity in an APP/DAPP problem refers to the need for intermediate placement operations for at least one part of the assembly. This means that in order for the problem to be solved some parts must be moved more than once. Because non-monotone (dis)assembly path plans require identification of intermediate

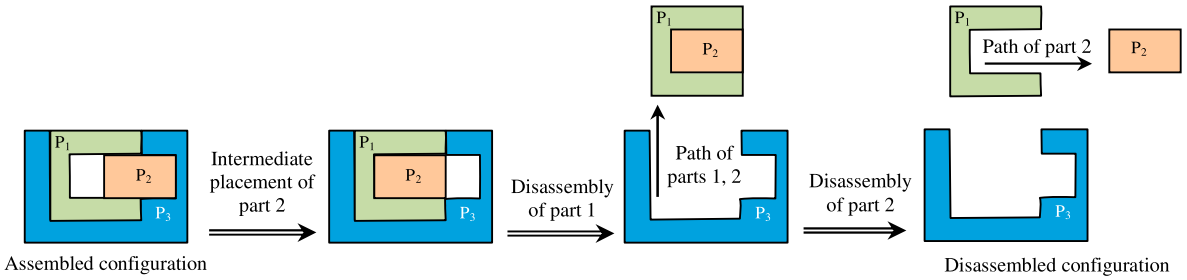

Fig. 6. A non-monotone disassembly path plan in which intermediate placement (i.e., temporary manipulation) of part P_2 is necessary for solving the problem. *Source:* From [91].

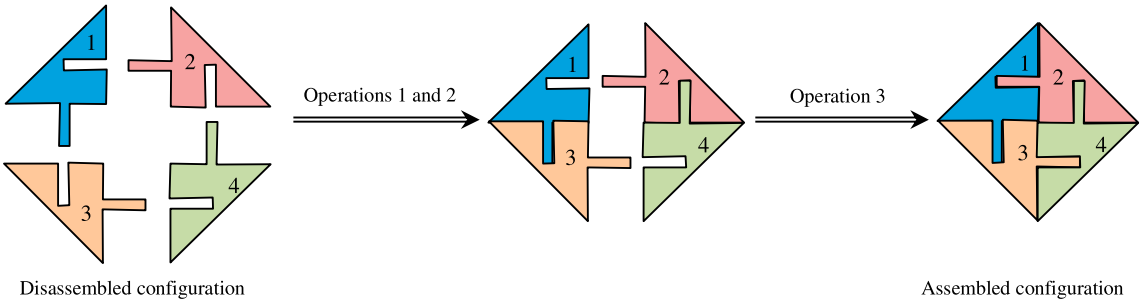

Fig. 7. A non-linear assembly path plan: the assembly Operation 3 involves inserting more than one part (i.e., subassemblies [1–4]) into the final assembly. *Source:* Inspired from [74].

positions for subassemblies, only a few (dis)assembly path planners have assumed non-monotonicity, including the works [58,71]. Therefore, most (dis)assembly path planners are applicable only to products with monotone (dis)assembly path plans. Fig. 6 illustrates a product with non-monotone disassembly plan.

2.2.4. Linearity

In a *Linear* APP problem all assembly operations involve inserting of a single part into the rest of the assembly, and in a linear DAPP problem all disassembly operations involve removing a single part from the rest of the assembly. Therefore, nonlinear (dis)assembly problems require simultaneous (removal)insertion of more than one part in at least one stage of the operation. Again many (dis)assembly path planners assume linearity in their problems since such an assumption simplifies the planning process considerably. Fig. 7 illustrates an assembly with nonlinear disassembly plan.

Only a few works have dealt with nonlinear APP/DAPP. For instance, the technique presented in [59] assumes non-linearity for disassembly plans. It automatically finds and clusters parts that can mutually affect each other's accessibility, forming 'part sets', and generates potential disassembly layers. Also, the methods presented in [67,71,74] are applicable to non-linear assembly path planning problems.

2.2.5. Coherence

Coherence in an APP/DAPP problem refers to the attachment of parts in a subassembly before being removed from/inserted into the assembly. In a coherent assembly, all parts inserted into the assembly must touch some previously-assembled part(s), and therefore is simpler than an incoherent assembly because the latter requires grasping or maintaining the stability of a subassembly before its assembly. A product with incoherent assembly plan is illustrated in Fig. 8.

To conclude this subsection, the features of the APP/DAPP Problem Nature along with their respective relevant articles are shown in Table 2.

3. Taxonomy of APP/DAPP solution methods

In this section, we present a taxonomy of APP/DAPP Solution Methods and survey the articles that have dealt with each feature of the taxonomy. As shown in Fig. 9, the taxonomy of APP/DAPP Solution Methods covers the two aspects of Approaches and Nature of solution methods.

3.1. Solution approaches

APP/DAPP solution methods can adopt different approaches in solving the problem, classifiable into five major categories of Graph-based, Grid-based, Sampling-based, Space Decomposition, and Interactive approaches. In fact, all these approaches adopt a unique methodology for coping with huge solution spaces of APP/DAPP problems: the Graph-based and Grid-based approaches do so by discretizing the space, the Sampling-based approach abstracts the space by sampling a limited number of points in it, the Space Decomposition approach divides the space into smaller subspaces, and in the Interactive approach human intelligence and interaction assists solving complex APP tasks.

Graph-based and *Grid-based* methods try to discretize the search space and thus reduce searching the whole space into searching a subspace of it. However, Graph-based methods construct graphs with vertices either dependent on the assembly parts, or in a space other than the C-space of the assembly parts, usually linked to the C-space via a vectorial or force mapping, whereas Grid-based methods discretize the C-space uniformly and search it for feasible paths of parts.

Most Graph-based and Grid-based Planners rely on explicit representation of free configurations in the C-space and become impractical when the dimension of the C-space grows. On the other hand, *Sampling-based* methods obtain an implicit representation of the C-space by checking the collision-freeness of a finite number of samples (points in the C-space) and thus do not suffer from the curse of dimensionality. Being currently the most practical path planners for high dimensional spaces, Sampling-based methods comprise a specific family of heuristic methods due to their similar methodology and wide applications. In the Sampling-based

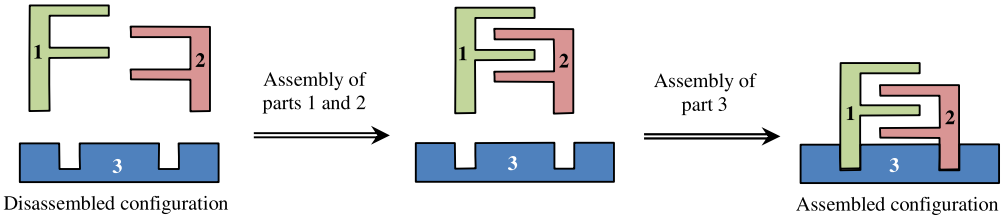

Fig. 8. An incoherent assembly path plan: parts 1 and 2 do not touch any previously-placed part (i.e., part 3) before being assembled.
Source: From [79].

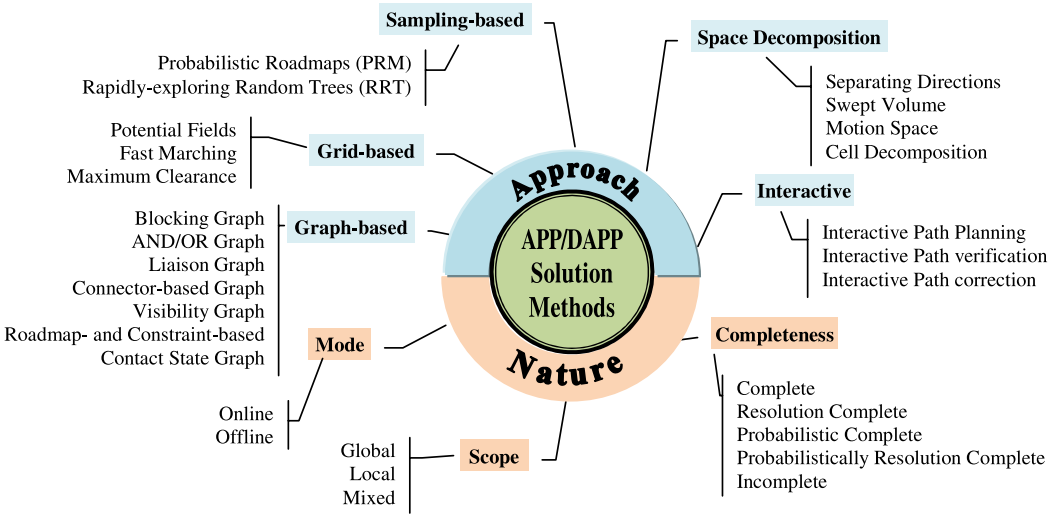

Fig. 9. Taxonomy of APP/DAPP solution methods.

Table 2
Relevant works in APP/DAPP Problem Nature features.

NATURE	**Scale**	**Gross** [40,41,56] **Fine** [13,17–20,31–33,35–39,42,45,46,48–55,57–90]
	Coherence	**Coherent** [17,18,32,33,36–40,42,46,48,50,51,56,57,59,61,63–65,67–73,76,77,79–83,85–90] **Incoherent** [13,19,20,31,35,41,45,49,52–55,58,60,62,66,74,75,78,84,90]
	Linearity	**Linear** [13,17–20,31–33,35–42,45,46,48,49,51–58,60–66,68–70,72,73,75–83,85–88,90] **Nonlinear** [50,59,67,71,74,84,89]
	Monotonicity	**Monotone** [13,17–19,31–33,35–42,45,46,48–57,59–70,72–90] **Non-Monotone** [20,58,71]
	Sequentiality	**Sequential** [13,17,18,31–33,35–40,42,45,46,48,49,51–57,59–64,66,68–82,84–88,90] **Non-Sequential** [19,20,41,50,58,65–67,83,89]

approach, numerous collision-free configurations (mostly random samples in the search space) for the parts or the assembler are generated and connected by some local free edges to form a graph (also called a *roadmap*), which is then searched to obtain a solution to the path planning problem. For APP/DAPP problems, the most commonly considered performance measures in Sampling-based methods have been the number of collision checks, planning time, and execution time [13,32,39,40,49,59,61–64,70,75,76,78], although objective functions like maximizing path smoothness have also been considered as performance measures [45,56].

The *Space Decomposition* approach adopts the Divide and Conquer methodology for simplifying the problem by dividing its solution space into subspaces (or subproblems) and then solving each subspace locally. Dividing may be done by motion directions separation, swept volumes, Motion space, or cell decomposition.

Interactive methods usually integrate the ASP and APP problems in an iterative and interactive way. These methods generally provide a modular architecture comprised of a number of modules with specific functions for solid model construction or acquisition, sequence planning, path planning, path optimizing, simulation, virtual reality, etc. The APP/DAPP problem is usually solved in an iterative manner through interaction with the assembly sequence planning module or a human user.

Most APP/DAPP methods have some common points with existing approaches to ASP, ALB, and Robot Motion Planning (RMP). Given an initial and a goal configuration of a robot A in a Euclidean space W with obstacles $B_1, \ldots, B_r$, the general RMP problem (also called the "Piano Movers' Problem") seeks a path τ for A and from its initial to goal configurations, while collisions with B_i are avoided [14]. Comprehensive reviews for RMP approaches exist in the books [15,92]. The sequential and monotone APP problem is similar to the general RMP problem since a particular part of the assembly can be considered as the robot A and the remaining parts of the assembly can be considered as obstacles. Hence most RMP methods are applicable to APP/DAPP. However, in the APP problem the initial (disassembled) configuration is not predefined explicitly as the parts are usually somewhere far from the assembly area, and the connectivity and precedence relations between assembly parts play an important role in planning their paths, while in the RMP the goal configuration must be specified and the relations between obstacles does not affect path planning. Also, in

Table 3
Main approaches in ASP, ALB, APP, and RMP Problems with their common points and differences.

Solution approach		ASP	ALB	APP	RMP
Graph-Based	Blocking graph	✓		✓	✓
	AND/OR graph	✓		✓	✓
	Liaison graph	✓	✓	✓	
	Connector-based graph	✓	✓	✓	
	Visibility graph			✓	✓
	Roadmap- and Constraint-based			✓	✓
	Contact State Graph	✓		✓	✓
	Precedence Graph	✓	✓		
	Voronoi Diagrams				✓
	Silhouette method				✓
	Subgoal network				✓
	Retract-like methods				✓
	Petri Nets	✓	✓		✓
Grid-Based	Potential fields			✓	✓
	Fast marching method			✓	✓
	Maximum clearance method			✓	
	Wavefront				✓
	Continuous Dijkstra	✓	✓		✓
Sampling-Based	Probabilistic Roadmaps (PRM)			✓	✓
	Rapidly exploring Random Trees (RRT)			✓	✓
	Randomized Path Planner (RPP)				✓
	Expansive-Spaces Trees (EST)				✓
	Sampling-based Roadmap of Trees (SRT)				✓
Space decomposition	Separating directions	✓		✓	✓
	Swept volume			✓	✓
	Motion space	✓		✓	✓
	Cell decomposition			✓	✓
Interactive	Interactive planning	✓	✓	✓	✓
	Interactive verification	✓	✓	✓	✓
	Interactive correction	✓	✓	✓	✓
Mathematical Programming	LP, NLP, IP, DP, GT, CT, etc.	✓	✓		✓
Metaheuristics	TS, SA, GA, ACO, PSO, AIS, etc.	✓	✓	✓	✓
Intelligent Computation	ANN, FL, FC, ES, LG, etc.	✓	✓		✓

RMP obstacles normally remain static whereas in APP the 'obstacle' (i.e., the assembly so far) is incrementally "growing" as the assembly proceeds, and therefore APP maybe considered as a collection of RMP problems. Due to such differences between APP/DAPP and RMP problems, a number of methods have been exclusively developed for the APP/DAPP problem, and correspondingly, some approaches are used only for RMP problems. Table 3 lists 33 main solution approaches in eight categories, together with their applicability to the ASP, ALB, APP, and RMP problems. The Table is compiled based on existing surveys, reviews, and textbooks on ASP [3], ALB [9], RMP [15,92], as well as our detailed study of the APP literature, and is a guide to identify common and unique solution approaches in these fields. The main approaches in solving APP/DAPP problems are described in the following.

3.1.1. Graph-based

Graph-based methods plan feasible (dis)assembly paths through three basic schemes: (1) mapping individual parts or a group of parts as nodes, and their mutual relations (in terms of connectivity, blocking, forces, constraints, etc.) as arcs into a graph (not in C-space), and finding appropriate paths using this graph, (2) constructing a graph based on geometrical features of parts (in C-space) and searching it for appropriate paths, and (3) representing all valid contact states between parts and inter-state relations as a graph and planning dis/assembly paths as sequences of contact state transitions. The first scheme contains the *Blocking Graph* (with some variations), *AND/OR Graph*, *Liaison Graph*, and *Connector-based Graph* methods, the second scheme includes the *Visibility Graph* and *Roadmap- and Constraint-based* methods, and the third scheme consists of the *Contact State Graph*, all described in this subsection.

3.1.1.1. Blocking graph. Blocking Graph was one of the first methods developed specifically for assembly sequence planning by R.H. Wilson in his dissertation at Stanford University in 1992 [79]. He invented a powerful tool called *Directional Blocking Graph* (DBG) with nodes corresponding to individual parts and arcs representing blocking relations between parts, so that an arc is plotted from P_i to P_j iff P_j blocks the translation of P_i along a particular direction d. Thus, if the parts of a subassembly form a subgraph in DBG of direction d without outgoing arcs, it means that no other part blocks the motion of that subassembly in direction d, and so there is a collision-free disassembly path for that subassembly in that direction (Fig. 10(a)). The notion of DBG was later used in developing tools like Local Constraint Digraph [69] and Directional Force Graph (DFG) [36] where labels of the arcs between any two nodes (parts) represent the maximum static force that has to be exerted to achieve an infinitesimal displacement between those parts along a specific direction.

Wilson and Latombe crafted the concept of *Non-Directional Blocking Graph* (NDBG) in 1994 by dividing the total space of assembly directions (which is a unit circle S^1 for 2D assemblies and a unit sphere S^2 for 3D assemblies) into subspaces of similar DBGs to present how the assembly parts block the movement of each other at all directions (Fig. 10(b)) [81]. Subsequently, many Exact methods for APP/DAPP problems employed the DBG and NDBG concepts. For example, in [68] DBG and NDBG were generalized for any

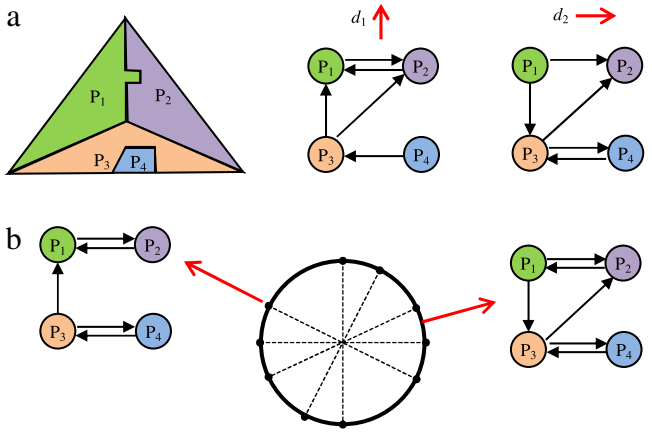

Fig. 10. (a) A simple product and two DBGs; (b) part of the NDBG of the assembly in (a).
Source: From [81].

translational and rotational movements between parts, which created blocking graphs in higher-dimensional configuration spaces for clearly showing the interactions between pairs of parts for any relative motion. Also, Latombe et al. extended the NDBG for products made of toleranced parts [19].

Wilson et al. in 1995 considered multi-step translations in 2D assembly path planning [33]. In that method, at first an interference diagram is created, and then a multi-step collision-free path for a subassembly is determined as a sequence of connected cells in this representation. The interference diagram is created by superimposing the Minkowski differences of all pairs of parts in the assembly with respect to the same reference point. As a result, some cells are created by intersection of the Minkowski regions, which are labeled with their respective colliding parts. The reference point is considered in the initial 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 DBG. Starting from the initial cell of a disassembly path for a subassembly, if an outgoing arc is added to a node of the desired subassembly in the DBG by traversing a particular cell, an alternate route has to be determined [3]. Rakshit and Akella in 2013 used the DBG concept to solve the Motion Stability Problem (introduced in Section 1) in the presence of physical forces such as gravity and friction [18]. As each part is removed from the assembly along a specified path, their method uses *linear complementarity* (developed in [93]) to analyze stability of parts in the remaining assembly. Since the motion of all parts in an assembly are tracked, instability-inducing motions can be identified and prevented by introducing appropriate fixtures, selecting alternative disassembly sequences, or changing the motion paths.

3.1.1.2. AND/OR graph. AND/OR Graphs were first used by Homem de Mello and Sanderson in 1986 for representing all possible assembly plans of a given product [84]. Nodes in an AND/OR graph correspond to assemblies, such that the root node represents the final assembly, and terminal nodes indicate single parts. The set of all possible disassembly operations from a given assembly are denoted by a number of 'hyperarcs' (pairs of associated arcs) drawn from the representative node of that assembly to two nodes corresponding to the successor subassemblies (or parts) resulting from removing a single part from the assembly (Fig. 11(a)). Thus the AND/OR graph representation encompasses all possible ways to assemble a product, and therefore allows one to search the space of all possible plans for finding the best assembly plan by means of the AO* search algorithm.

3.1.1.3. Liaison graph. In the Liaison Graph representation of an assembly, nodes correspond to individual parts and edges represent contact relations between parts such that two parts are connected iff one of them constrains the freedom of motion of the other either by a direct contact or a near contact (i.e. having a distance smaller than a predefined threshold), as shown in Fig. 11(b). Lee and Shin [50] presented a method for automatic determination of preferred assembly partial orders using liaison graph, in which the feasibility of an assembly operation is checked by calculating directional freedom of motion, manipulability, interference with neighboring parts, and accessibility of a part. Determining an assembly partial order is based on recursive extraction of preferred subassemblies and verification of their disassemblability. A preferred subassembly is a cluster of parts which are naturally and easily disassemblable from the original assembly incurring minimal assembly cost, and can be selected by evaluating tentative subassemblies based on Stability Index (SI) and Structural Preference Index (SPI) which respectively measure the mobilities and structural complexities within and between clusters.

3.1.1.4. Connector-based graph. Tseng and Li developed the Connector-based assembly sequence graph approach for generating assembly sequences and paths based on various types of parts connectors (such as screw, bolt-nut-washer, pin fit, taper fit, spring fit, rivet, ring, bearing, weld, glue, etc.) and their representation schemes [89]. An assembly is decomposed into a set of connector-based assembly elements, and a Connector-based assembly graph is created based on precedence relations between assembly parts (Fig. 12). This graph can be either single-path or multi-path: in a single-path connector-based graph, first a connector-based assembly element is selected and added to the assembly sequence, and then the remaining elements are incrementally added to the assembly sequence based on the interference analysis between the selected element and the elements already existing in the sequence. In the multi-path connector-based graph, any element C^i can be moved out of the assembly sequence and a new path parallel to the original one is created only if C^i has no interference with its immediate predecessor and successor elements in the Connector-based graph, and there exists a successive element to which C^i can be linked.

3.1.1.5. Visibility graph. The Visibility Graph (VG) method was first proposed by Nilsson in 1969 and has found wide applications in robot path planning [94]. For solving an APP/DAPP problem, the VG constructs a roadmap that preserves the shape and connectivity of the free space between assembly parts, thus converting the continuous workspace of the (dis)assembly into a graph-like structure. The VG method is more appropriate for assemblies that have a few polygonal parts, where vertices of the polygons are the nodes of the VG and there is an edge connecting any two nodes if their corresponding vertices are within the line of sight of each other, as illustrated in Fig. 13(a). The VG method was used in 1994 to find an initial separation direction of a free part relative to a reference stationary subassembly in planning a collision-free disassembly path for that part. The magnitude of the separating translation was determined by projecting the vertices of both components (the part and subassembly) onto the separation direction vector [37]. Visibility maps have also been used to reduce the computational time and memory of aircraft assembly path planning by taking into consideration both assembly mating features and assembly constraints in interference checking of two mating parts [82].

The Visibility Graph has also been used as a tool for path planning of parts from their initial to final configurations. In [35], after constructing the VG of the parts in the C-space (obtained by calculating Minkowski Sums of parts) using the revolve scanning method, some 'selected set' of feasible assembly paths are found,

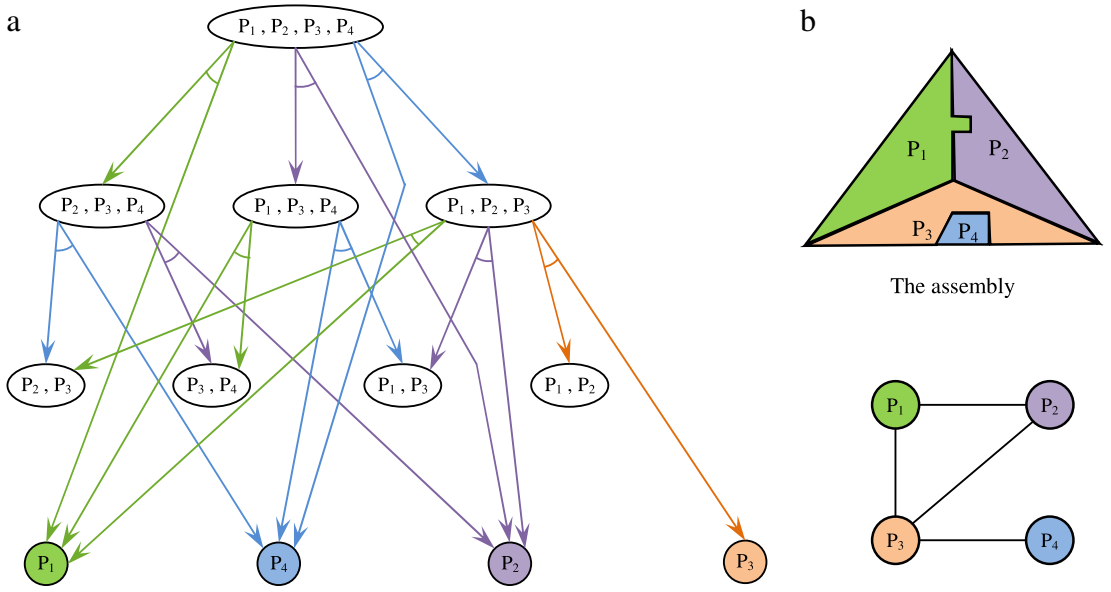

Fig. 11. (a) AND/OR Graph, and (b) Liaison Graph representations of a simple assembly.

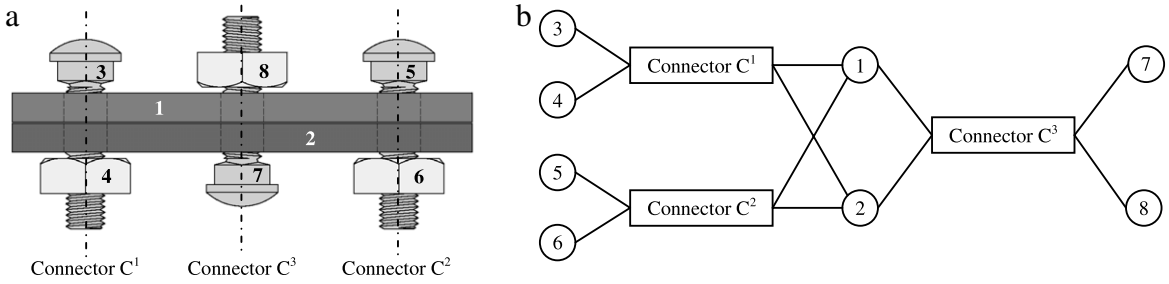

Fig. 12. (a) Two plates connected by three bolts; (b) Connector-based assembly sequence graph.

and finally the Ant Colony Optimization (ACO) algorithm is implemented for finding reasonable paths due to its concurrence, high efficiency, and the mechanism of positive feedback, with less search time compared to the Dijkstra search.

3.1.1.6. Roadmap- and constraint-based. Garber and Lin presented a framework applicable for the Maintainability Problem (introduced in Section 1) [66]. They reformulated the motion planning problem as a simulation of a constrained dynamical system, which is then guided by the Generalized Voronoi Diagram (GVD) (Fig. 13(b)). The robot that manipulates the part to be maintained must stay maximally clear of nearby obstacles and satisfy all constraints that are locally imposed on it. The constraints are classified into two categories: *hard constraints*, which must be absolutely satisfied through the planning (like object non-penetration, joint connectivity, and angle limits on the joints of an articulated robot), and *soft constraints*, which guide the parts to behave in a desired way (like moving along paths designated by the GVD, moving toward a goal, and avoiding nearest obstacles). Penalty-based methods are used to handle soft constraints and iterative relaxation is used to enforce hard constraints.

3.1.1.7. Contact state graph. The Contact State Graph method was presented in [46] to model different types of contact states and contact and friction forces of a deformable elastic tube during virtual assembly. In the Contact State Graph each node represents a valid contact state between two parts, and each arc between two nodes implies the adjacency of their corresponding contact states. Based on this graph, relative assembly paths can be planned as sequences of contact state transitions connecting the starting state to the goal state.

3.1.2. Grid-based

Grid-based methods locally explore a uniform regular grid in the C-space and find appropriate paths for assembly parts. These methods are generally resolution-complete, meaning that they are guaranteed to find a solution if the grid resolution is sufficiently fine. The Potential Fields method covers a family of similar works, all based on goal-directed gradient search. On the other hand, other graph search techniques like Fast Marching Method and Maximum Clearance Method have also been used for planning obstacle-free paths on the grid.

3.1.2.1. Potential fields. The Potential Fields (PF) method was first developed in 1986 by Khatib [96] in which a robot is treated as a point in the configuration space being under the influence of an artificial potential field $U = U_{att} + U_{rep}$, where U_{att} is the attractive potential of the goal and U_{rep} is the sum of repulsive potentials exerted by nearby obstacles. The robot then follows the steepest gradient from its current grid node until the goal, which has the lowest potential, is reached. Despite its efficiency and extendibility to high-dimensional spaces, the original Potential Fields method suffered from trapping in local minima.

While the PF method has been widely used in robotics for local motion planning and obstacle avoidance, there are relatively few works employing this method in APP/DAPP. Gottschlich and Kak

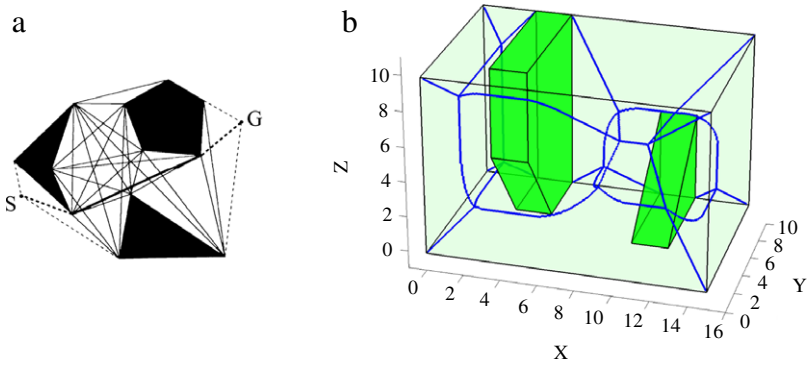

Fig. 13. (a) A Visibility Graph connecting start (S) and goal (G) points via a path among obstacles [14]; (b) A Generalized Voronoi Graph in 3D which can be used for planning safe paths [95].

developed an assembly motion planning system called AMP-CAD that given CAD models of the parts and a description of the assembly operation, automatically finds the assembly motions of parts in two phases: in Phase 1, both uncertainties and contacts between parts are ignored and a collision-free path for the assembly operation is found based on a graph search over a Potential Field representation of parts. Uncertainties and potential contacts between parts are then considered in Phase 2 in which the paths generated in the Phase 1 are analyzed for likelihood of collisions. Afterward, in all path segments that are in collision, compliance is introduced through force/torque guided motions. The modified assembly path plan is then integrated with automatic error detection and recovery (EDR) procedures and sent to the execution unit [48].

Hassan and Yoon developed a framework based on PF and ACO for automated assembly process, which is considered the most difficult phase in virtual aircraft manufacturing [52]. For each handled part, all collision-free paths are first generated using the PF and then the ACO algorithm is used to find the shortest of the assembly paths. In another article, Hassan and Yoon developed an assembly planner with a combined approach based on the PF and Genetic Algorithm (GA) methods for path planning in a virtual Maintenance Assembly/Disassembly (MAD) environment [53]. The proposed approach is performed in three levels: Initially, 3D CAD models of parts with their initial and final positions are loaded, and in level 1 all valid (dis)assembly sequences and the number of gripper changes for each sequence are calculated and stored in the Sequence Population (SP) set. Then, in level 2, the PF method is used to plan a path for all parts of each sequence in the SP. Successful paths are then stored in the Path Population (PP) set. Once SP and PP sets are completed, GA is implemented to generate the near-optimal sequence, which is then used to plan near-optimal paths for parts in that sequence. Finally, in level 3 the passive haptic control mode is applied to the virtual environment and the obtained paths are corrected by the user. Yoon also used the PF method in [51] for assembly path planning of parts to their goals in an assembly, which along with automatic assembly sequence planning, are used to compute initial assembly paths and guide human operators in finalizing the paths through haptic guidance forces. The PF method was also used in a path planner in conjunction with the Fast Marching Method [55] as described in the next subsection.

In order to eliminate the local minimum problem, Chang and Li developed a planner based on Randomized Path Planning (RPP) which used PF as a guide to search for a disassembly path in a Maintainability Problem, but in order to escape from local minima they used a preset number of random walks (Brownian motions) each of which was followed by a gradient motion. In case that all these attempts fail, a backtracking step is performed to retract the part from the path found so far [31]. In another article, the same authors extended their work by planning access paths through some 'preferred regions', which satisfy realistic concerns in assembling like the size of the moving part, support regions for tools, heat sources to avoid, or simply access convenience. They built an interface for the user to specify such constraints which are called 'translational constraints' by confining the moving part in a 'constraint volume'. This volume is computed from a sequence of spheres with various diameters connected smoothly by trimmed cones. In addition to translational constraints, there could be 'rotational constraints' that need to be taken into account in maintainability studies. These constraints typically can be described as limitations on the rotations about certain axes, called the constraint axes. To meet rotational constraints, a new local reference frame F^c is created where one of its axes is aligned with the constraint axis [54]. Other methods of coping with the local minimum issue include using available global information for escaping from local minima [48], performing a gradient motion [31,52], and calculating minimum and maximum bounding boxes for the parts and obstacles [51].

3.1.2.2. Fast marching method. The Fast Marching Method (FMM) is a computational technique that numerically approximates the solution to the Eikonal nonlinear partial differential equation $|\nabla u(x)| = F(x) > 0$ in $\Omega \in \Re^m$ ($m = 2$ or 3), and is a very efficient, robust, and accurate method for searching rectanguloid grids [97].Lee et al. presented an active vision system for micro-assembly path planning. The system was equipped with a fixed top view camera and a side view camera with a rotational mirror system for freely changing the viewing direction. They used the FMM to generate a safe path in two image spaces and employed a visual tracking method to follow this path in the image space. In order to avoid visual occlusions and obstacles they used a Potential Field method to build the travel cost for the FMM. Also, epipolar boundary constraints were considered in the potential field generation for efficient path planning in image spaces [55].

3.1.2.3. Maximum clearance method. A recent grid-based method for automatic assembly path planning of electric cables and wiring harnesses in automotive industry is presented in [45], in which it is assumed that a wiring harness can be manipulated by moving a set of grip handles in order to cope with challenges related to the concealed routing and multiple branching points. At first, given a harness mounted at its target configuration, a 'master handle' is attached to a central point, and all other handle constraints are relaxed. Then through a two-step path planning procedure, first a local search is performed to find the maximum possible clearance of the master handle from the set of obstacles, and then a standard A* search is used to find the shortest collision-free path for the master handle on a grid in $\Re^3$. After generating a disassembly

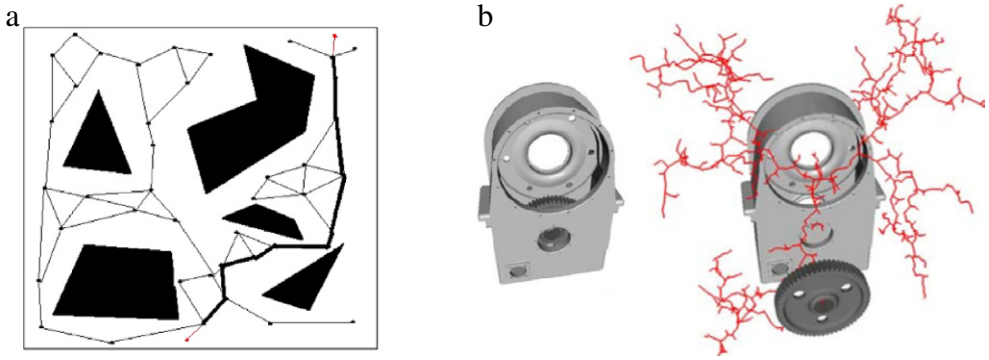

Fig. 14. (a) A Probabilistic Roadmap (PRM) is a graph of random nodes connected by straight edges [100] which is searched for the shortest path; (b) A Rapidly-exploring Random Tree (RRT) for extracting a gear from its casing [32].

path, the harness is unfolded wherever the free space around provides sufficient freedom, and then the paths are reversed and locally smoothed for each handle. Finally, possible existing contacts during the manipulation are resolved by attaching a set of supplementary handles to the harness layout.

3.1.3. Sampling-based

Since the size of the composite C-space grows exponentially with the number of parts of an assembly, its explicit construction and searching is impractical for high-dimensional spaces under complex constraints. The Sampling-based approach was developed in mid 1990s to remedy this issue, and is founded on abstracting the space by selecting a limited number of points in the C-space and connecting them to form a roadmap. The main challenge in sampling-based methods is capturing the connectivity of the C-space (especially through narrow passages) and securing a connected path between the start and goal configurations. Early sampling-based path planning methods were developed by Kavraki et al. in 1996 (Probabilistic Roadmaps, PRM) [98] and LaValle in 1998 (Rapidly-exploring Random Trees, RRT) [99], which create cyclic and acyclic random graphs, respectively. Fig. 14(a) and (b) show typical PRM and RRT graphs. Despite their efficiency and ease of implementation, sampling-based methods are probabilistic-complete, meaning that they guarantee to find a feasible path (or report that no such a path exists) within infinite time. Sampling-based methods have been applied for solving APP/DAPP problems since 2000.

Some performance measures have been used to demonstrate the efficiency of sampling-based planners, including: minimizing the number of nodes in the random roadmap [39,61,63,70,75,76], minimizing the number of collision checks [13,39,63,75,78], minimizing the number of total random samples [13,39,49,63], minimizing the number of edges in the random roadmap [76], minimizing the number of contact queries [13], minimizing the number of iterations [58,70], minimizing the number of failures [78]; maximizing the success rate [63,70,75], and generating sufficient non-colliding samples in narrow passages [13].

3.1.3.1. Probabilistic roadmaps (PRM). Sundaram et al. used the PRM approach for solving the DAPP problem in [76]. The parts in the assembly are considered as robots and a composite configuration space is formed from the individual configuration spaces of parts, in which the PRM method takes random samples. Because of the high dimension of the composite C-space and the narrow passage problem, the sampling is biased by computing some potential movement directions such as directions perpendicular to part faces based on the geometric characteristics of reachable configurations from the assembled (starting) configuration. The use of the face normals as potential removal directions was inspired from the *casting problem* in computational geometry, which has applications in manufacturing as it involves removing a cast part from a mold [101].

Thomas and Iser in [78] proposed a probabilistic path planning algorithm based on the PRM algorithm applicable for APP/DAPP problems, which is able to find paths even when many narrow passages exist. The proposed path planner propagates new samples, discards samples evaluated as 'bad' and assigns higher weights to 'good' samples. The evaluation function propagates the samples to explore the free space and concentrate on borders of obstacles, which helps samples to locate inside narrow passages.

In 2013 Carlson et al. presented a method that first generates an assembly path by the Lazy PRM method [102] which does collision checks only when it is absolutely necessary, and then post-processes the generated path by some smoothing algorithms (including simple interpolation, gradient-based interpolation, and gradient-based cubic polynomial) which try to minimize the probability of collision. In order to modify the distance measure needed to smooth the initial path, a variation analysis is performed. The existing uncertainties stem from geometrical variations in the environment and in the robot [56].

3.1.3.2. Rapidly-exploring random trees (RRT). Perhaps RRT is the most implemented method in APP/DAPP problems, starting two years after its inception. Vallejo et al. in 2000 proposed an adaptive framework for 'single shot' motion planning, i.e., planning without pre-processing, which is applicable to the Maintainability Problem where the robot's free C-space has narrow passages [62]. They incorporated several planners grouped in four sets so that a query could be matched with the best planner for it: (1) simple local planning methods such as straight line in C-space, rotate-at-*s* (for rigid bodies), and simple A*-like planners, all introduced in [103] (2) directed expansion methods such as Iterative Spread Method (ISM), Iterative Translational Method (ITM), Iterative Rotational Method (IRM), and Ariadne's Clew Algorithm (ACA) introduced in [104], (3) random expansion methods such as Random Walk (RWM) and RRT, (4) subgoal generation methods such as First Intersection Method (FIM) and Recursive Midpoint Method (RMM). The 'single shot' method was compared to Obstacle based-PRM, RRT, and ACA methods and outperformed them in runtime and ability to solve complex problems.

Sampling-based methods rely heavily on collision detection procedures, and so Ferré and Laumond introduced an RRT-based efficient path planner in [64] to reduce the cost of collision detection, which computes an initial path while allowing some penetration within the obstacles. The obtained paths are then re-shaped iteratively by decreasing the allowed penetration threshold. When this procedure fails, the search restarts with a roadmap composed of the collision-free portions of the path. Later, Ferré et al. in [40]

used this planner to find paths of parts disassembled by an articulated mannequin: a part is first considered as a free-flying object and then inverse kinematics operators compute the motions of the mannequin. In case of failure, the composite C- space is searched for finding feasible paths for both the mannequin and the part.

Aguinaga et al. in [32] presented two automatic planners to solve the *Selective Disassembly Planning* problem (introduced in Section 1) in a virtual environment, which aims to disassemble a specific part. The first one is based on single translations, while the second is based on the RRT approach. The same authors later introduced the Targetless RRT (T-RRT) algorithm for solving the Selective Disassembly Planning problem [61]. Their proposed disassembly planner uses only geometric information in the form of triangle meshes as its input, and the procedure for generating the precedence graph is based on removal of exterior parts of the assembly, layer-by-layer and proceeding inwards, until the target part is removed. The removal of a part cannot occur until all the parts that limit its movement have been removed. They used two space decomposition techniques based on voxels and octrees for auxiliary algorithms such as collision detection or localization of the exterior parts of an assembly.

Cortés et al. [39] presented the Manhattan-like RRT (ML-RRT) algorithm for finding solution pathways for articulated objects, which can be generalized to the protein ligand interactions problem (for computing an exit pathway of a ligand from a deep active site to the surface of a protein, or vice versa). All articulated objects are considered 'mobile' as opposed to 'static' obstacles (for example in Fig. 3(c) the red/dark I-shaped and the three blue/light 'sticks' are mobile objects, whereas the gray surrounding object is static). The ML-RRT algorithm alternately plans paths for the object to be disassembled and for the hindering mobile objects. During the path planning process, mobile objects may be either *active* or *passive*, depending on the need for planning their paths. Finally, a randomized path-smoothing post-processing is performed in the composite configuration space of all parts so that simultaneous motions are obtained in the final path.

In order to improve the performance of the original RRT, an approach called D-plan is presented in [13]. This approach contains two techniques: the first is an optimization-based retraction algorithm [105] to generate samples in narrow passages and near the boundary of the C-obstacle space, and the second is a constraint-based motion interpolation algorithm that effectively connects nearby samples by taking into account the non-collision constraint for the closest features between the robot and obstacles.

A new method for simultaneously (dis)assembly sequence planning and path planning called Iterated Manhattan-like RRT (I-ML-RRT) algorithm was introduced in [58], which involves rapid computation of paths for extracting all parts from an assembly. At each iteration of the I-ML-RRT algorithm one part is tried to be disassembled. Part selection may either be a random process or follow a predefined disassembly order determined by a heuristic based on the assembly structure. This algorithm can handle non-monotone disassembly sequences. A path planning algorithm called Danger-zones RRT (DRRT) was presented in 2010 for planning feasible (dis)assembly paths in an environment with 'danger zones', which though not being forbidden (i.e., blocked by obstacles), are undesirable areas in the C-space that must be avoided as much as possible [63]. Danger zones exert soft constraints, and generating a path through them is not desirable, but acceptable if no better path exists or can be computed efficiently. The algorithm utilizes a sampling function based on kurtosis coefficient, which is the fourth standardized moment defined by μ_4/σ^4 where μ_4 is the fourth moment about the mean (a measure of the 'peakedness' of the probability distribution of a real-valued random variable) and σ is the standard deviation, for planning in the free C-space that avoids high density areas of tree nodes. Identifying dense areas is done using a density measure proposed in [106]. Also, a random sampling function was introduced for obtaining random states from the triangles that describe the danger zones.

Recently, in 2013, the RRT has been employed in three relevant works. For APP of complex structures an Obstacle and Greedy rule-based RRT (OG-RRT) algorithm is proposed in [70], which first guides the extended direction of the random tree with the collision information (i.e., coordinates and normal vector at collision point) of the target part and obstacles, and then extends nodes by a greedy rule along each direction with translation-after-rotation expansion strategy. Finally, piecewise linear approximation is used to optimize the original assembly path. In [59] a technique is presented that combines motion planning and part interaction clusters to improve generation of assembly precedence constraints. The proposed technique automatically finds and clusters parts that can mutually affect each other's accessibility, and hence may impose assembly constraints. Potential disassembly part sets are generated using spatial clustering. Next, the RRT algorithm with multiple trees is used to evaluate the interaction between these part sets and determine the part sets that can be removed from the assembly. These sets are added to the first disassembly layer and removed from the assembly. Part sets that can be removed from the simplified assembly are then added to the second layer. If the process gets stuck, parts in the parent set are regrouped and the process continues until all disassembly layers are found. The resulting structure reveals precedence relationships among part sets, which can be used to generate feasible assembly sequences for each part set and the whole assembly. In [75] assembly paths are automatically generated in complex 3D environments through a strategy biasing the RRT tree based on its planning history. The global path planning process is divided into three phases and specific algorithms are proposed in each phase: in the first phase a stochastic collision detection method is proposed for evaluating the intersections between two polyhedral parts, and a refined history-based RRT which biases the tree toward unexplored areas is used in the second phase. Finally in the third phase a novel adaptive RRT algorithm is used which assigns values on each tree node in order to explore complex environments more efficiently.

3.1.4. Space decomposition

Space decomposition methods are usually based on a simplification of the workspace through transforming or decomposing the initial workspace into simpler subspaces by spatial division of the search space. After the division, (dis)assembly paths of parts are found in all subspaces and concatenated to form final (dis)assembly paths. Space Decomposition methods include separating motion directions, swept volumes, Motion space, and Cell Decomposition.

3.1.4.1. Separating directions. A method for coping with large solution spaces is through Divide and Conquer approach; that is, to divide the space into subspaces, search each one, and integrate the answers. For solving APP/DAPP problems, a number of algorithms compute representative separating motion directions of the parts of an assembly, such as local translational freedom cones computed on the unit sphere S^2 for translating polyhedrons [73,74], and local depart spaces as 3D polyhedral convex cones [71]. For infinitesimal translations in 2D, each possible path for a part is defined by a single angular parameter representing the direction of the translation. Therefore, edge-edge contacts between polygonal parts induce a partition of the unit circle of directions S^1 in $\Re^2$. Likewise, for infinitesimal translations in 3D, each possible path for a part is defined by two angular parameters representing direction of the translation. Therefore, point–plane contacts between polyhedral parts induce a partition of the unit sphere S^2 in $\Re^3$ [81]. For infinitesimal generalized motions in 3D the set of all possible direc-

tions of motion make up the 5D unit sphere S^5, and so contacts between the polyhedral parts induce a partition of S^5 into the closed hemispheres [67].

Srinivasan et al. used geometrical information such as mating faces of parts to determine the disassemblability of a particular part of an assembly [20]. In that method, all disassembly directions of a part with respect to its mating faces with other parts are mapped onto a Gaussian circle (sphere) for 2D (3D) assemblies. Then the intersection of feasible disassembly directions for all mating faces yields the resultant disassembly direction(s) for that part. Gao et al. presented an interference-detecting method for determining the disassembly path of a particular part p_k in an assembly [57]. First, the bounding box contour of the whole assembly is calculated using the initial configurations of all parts. Then, based on the geometry, constraints, and other information of the assembly, the set S of candidate feasible disassembly (separating) directions and the set Q of the parts that block the motion of p_k along the directions in S are generated through detecting interferences of the parts. Then the part p_k is moved along the direction S_i ($i = 0, 1, \ldots, m-1$) with a rational step size l. At each step of the motion, through a four-step procedure, interferences between p_k and all the parts in Q are checked: in case of no interferences, a collision-free disassembly path is found for the part p_k along the direction S_i from its initial position toward outside the bounding box of the whole assembly.

The Disassembly Weighted Hybrid Graph (DWHG) is proposed in [90] based on CAD model of parts to describe the constraints and disassembly priority relationships among components of a product. The location of the target component for the Selective Disassembly Planning Problem (introduced in Section 1) can be determined based on the DWHG and its construction process. After generating initial feasible disassembly sequences, the PSO algorithm is used to obtain near-optimal disassembly sequences. Then the disassembly path of the target component is translated into the discrete critical path points and its set of all feasible local disassembly directions is determined by the intersection of all possible directions derived from the mating parts of the target component and features at each point.

Recently, in 2014, a new approach has been introduced for automatic generation of exploded views of assembly parts from an assembly model [88]. In order to represent assembly relations and interactions among constitutive parts and their disassembly directions from the center of an assembly, a relational matrix called Extended Interference Matrix (EIM) is constructed from the assembly CAD model, which denotes the blocking relations of parts along some selected directions. Then for generating the assembly sequence and its related assembly paths, the part with the most connecting relations with other parts is selected iteratively based on the EIM, which results in a feasible sequence of parts. Then for each part, all possible separating directions are searched and selected, and the values in the EIM are modified for the exploded (separated) component or part. Finally, exploded views are created based on the assembly sequences and the information in EIM.

3.1.4.2. Swept volume. Some other Spatial Division methods form a swept volume of each part along its kinetic orientation in order to plan assembly paths. In [38] first an assembly sequence is planned using a hybrid Genetic Algorithm and Ant colony Algorithm (GAAA), based on which a Boundary Representation (B-Rep) filling algorithm which can form a swept volume between the part and its kinetic orientation is developed to plan assembly paths. In order to avoid assembly collisions during the product assembly design the assembly path is analyzed and may be sent back for replanning a better assembly sequence. In [72] a space swept algorithm is used for creating a swept volume to find effective assembly paths for aircraft parts. First an assembly sequence is planned by taking into account the fixtures and tools used in aircraft assembling, which serves as a guide to assembly path planning. The paths are then planned by means of the space swept algorithm.

Recently, in 2013, Popescu and Iacob presented a disassembly method based on two new elements: (1) Connection Interface concept, used for generating a set of connection interfaces for a product in order to define mechanical joints and automating the identification of fasteners in the assembly, and (2) Unit Ball concept, used for determining the parts' mobility within assembly [87]. The Swept Volume approach was then used to determine disassembly collision-free paths for assembly parts.

3.1.4.3. Motion space. As mentioned earlier, a common method for solving the (dis)assembly path planning problem for assemblies with many parts is to treat each part as an independent free-flying object and generate a path in the composite configuration space of the parts [14]. But the number of dimensions of the space grows linearly with the number of parts. A method to overcome this problem is the *Motion Space* method that uses the NDBG concept to accommodate more general part motions. This methodology is comparable to the notion of C-space in path planning; but while each point in the Configuration Space represents a possible *placement* of the objects, each possible *motion* of a subassembly is represented as a point in the Motion Space (M-space), which is determined by the NDBG at that point. All points in the M-space with equal NDBGs form a distinct subspace with a unique (dis)assembly scheme. The main advantage of the Motion Space method is that the number of dimensions of the M-space is independent of the number of parts in the assembly; however, it grows with the complexity of allowable motions. Here the complexity of a motion is measured in terms of its type (translational, rotational, or both), scope (infinitesimal, finite, infinite), direction changes (one-step, multi-step), etc. Through the Motion Space method, efficient paths for assembly parts can be found when the allowable motions can be described by relatively few parameters [17]. Fogel and Halperin also used the Motion Space for solving the Partitioning Problem (introduced in Section 1) in $\Re^3$ with infinite translations [65]. Their proposed algorithm consists of eight phases that exploit arrangements of geodesic arcs embedded on the unit sphere in various ways.

For two-handed coherent assemblies, a concept similar to the Motion Space was presented in [77], where the separability of an assembly of n parts into two subassemblies is computed by calculating the configuration obstacles through Minkowski differences of each pair of parts in the subassemblies. Then the C-obstacles are projected onto the unit sphere via $n(n-1)/2$ stereographic projections. For evaluating the subassemblies, an AND/OR-graph for the separability of all possible subassemblies is constructed and accurately searched.

3.1.4.4. Cell decomposition. Cell Decomposition is one of the first methods used in robot path planning, and is classified into two main groups of *Exact* and *Approximate* cell decomposition. Ladeveze et al. in [49] presented a global interactive framework including a Rapidly-exploring Deterministic Tree (RDT) algorithm and a real-time guiding force for 3D CAD part assembly or disassembly tasks. Path planning is performed in two steps: the first consists building a discrete representation of the workspace using an unbalanced octal tree (octree) (introduced by Jackins and Tanimoto in 1980 [107]) with an adjustable level of depth (i.e., resolution) depending on the size of handled parts in order to minimize the computation time and the memory space needed. In the second step, a 3-dimensional (including translations along x, y, z) volume path from the initial to the final position is computed using an A* algorithm based on octree data structure in order to accelerate the RDT convergence. Finally the RDT algorithm is used to find

a 6-dimensional (including translations along x, y, z, and rotations about θ_x, θ_y, θ_z) collision-free assembly path in real-time for the handled part using the 3D solution path generated by the A* algorithm. Once the path has been found, a haptic artificial guidance on the path is provided to the user.

3.1.5. Interactive

Interactive APP/DAPP methods typically rely on human aid in planning, verifying, or correcting of (dis)assembly paths. Interaction with human user is done through Virtual Reality (VR) environments or haptic interfaces. VR systems can be used to simulate, analyze, optimize and increase the efficiency of assembly planning simulations [108]. These systems allow more realistic human–computer interactions and have become increasingly popular for engineering applications such as CAD and process evaluation. Since physical objects are represented as virtual objects in a VR environment, real objects are not necessary for simulation of a process, and thus VR systems dramatically reduce the time and cost of an assembly process. On the other hand, haptic technologies enable a human user to feel the force feedback from the virtual environment, leading to a more intuitive and natural way to simulate the assembly process during the design phase of new components even before any physical prototype is created [43]. Virtual disassembly simulation of varying target components for complex products becomes more essential when disassembly planning and analysis focus on changing disassembly demand (for example maintenance or precious parts recycling) [90]. One of the major but less-known advantages of VR technology is related to *data logging*. User logging records provide rich data for automatically generating designs or manufacturing instructions. Also, extrapolating cognitive procedures relating to assembly tasks during user interaction will provide information for performing a better product Design For Manufacture and Assembly (DFMA) [44].

3.1.5.1. Interactive path planning. Jin et al. used virtual assembly technology to perform assembly path planning of a mechanical decelerator. At first, 3D solid models of components of the decelerator are built based on their structural features, and then a partial assembly is created by grouping all components. Afterward, various kinds of assembly relations are hierarchically added between components. Finally the assembly simulation is completed by creating assembly paths of components through manual planning realized by human interaction [83].

Siddique and Rosen combined automated and interactive techniques using Virtual Prototyping to generate complete disassembly processes of a product design. The partial disassembly sequences and the range of removal directions (paths) for all parts are generated automatically and then based on this information and the geometry of the components, the designer decides on the component or fastener to be disassembled and the direction of removing them by interactively disassembling the product in the virtual environment [42].

A feature-based assembly model was proposed in 2002 for automatic disassembly sequence and path planning based on geometric reasoning and knowledge [85]. The information provided by the mating features of parts in the product was used to find the candidate parts for disassembly and to carry out disassembly path planning. Also, an interference-checking approach was used to ensure no global collision while disassembling a part. A set of criteria and heuristic rules based on knowledge, constraints, relationships among parts, and quantitative disassemblability assessment were used to implement the model more efficiently. The model can also be used interactively by a user when necessary, enabling him to visually disassemble the product while planning.

Yan et al. presented an interactive path planner for virtual assembly and disassembly operations [60]. At first, the user manipulates mechanical parts by a force feedback haptic device with some penetration into obstacles. The generated rough path is retracted by a random retraction method and then connected using an RRT-connect algorithm. For each operation of the user, a path successfully passing through narrow passages is presented, or in case of failure, the nearest configuration-pair in workspace is returned. The interference information is obtained from the contact between the configuration-pair and obstacles, and finally the user adjusts movements or modifies the designed size of parts according to the information.

3.1.5.2. Interactive path verification. An assembly planning and simulation system called AutoAssem was developed by Xu et al. in [80] to automate assembly planning for complex products such as aircraft components. The procedure involves the following steps: Assembly modeling based on object-oriented methods; ASP by a set of metaheuristic algorithms such as GA, ACO, PSO, or human–computer interaction; APP by Filtering algorithm and using the exploded view obtained from the assembly sequence and geometric information of the parts; Process planning and visualization based on the sequence of assemblies; and Assembly simulation which provides a means to the user for performing dynamic gap analysis to verify and determine necessary modifications to the assembly sequence or generated paths.

In another article in 2013, the same authors proposed object-oriented templates with various assembly relations to deal with the complexity of assemblies of large-scale products such as aero-engines [86]. The object classes represent the constitutive parts or subassemblies in a product structure, and the parameters of object classes represent the features of parts or subassemblies that can be defined or modified to meet the requirements of a specific instance. Then an object-oriented product assembly model is created based on the information of CAD system, product design documentation, and human–computer interactive input. The presented digital assembly system effectively supports functional modules including assembly planning (ASP and APP), assembly simulation, assembly process planning, and assembly evaluation and analysis.

3.1.5.3. Interactive path correction. In the works mentioned below, human interaction is sought to correct assembly paths automatically planned by the Potential Fields method described in Section 3.1.2.1, and thus we have enlisted them under the PF methods, but emphasize their human interaction aspect here.

Hassan and Yoon in 2010 proposed two methods for aircraft maintenance in a virtual assembling environment [52,53]. For each handled part, collision-free shortest assembly paths are first generated using Potential Fields, Ant Colony Optimization, or Genetic Algorithms, and a trail of these paths are then made visible in the assembly simulation environment which provide guidance to operators through a Sensible© PHANTOM™ haptic device (a 3-DOF device that provides six inputs in the form of force and torque coordinates and three outputs as force coordinates). A year later, Yoon worked on a virtual assembly system with haptic interactions during assembly operations. A haptic guidance force guides the operator to move parts from their initial to final positions along a prescribed path preplanned by the Potential Fields method, and also helps the operator in selecting proper grippers and assembly orientations [51].

The main features of APP/DAPP Solutions Approaches along with their respective relevant articles are shown in Table 4.

3.2. Nature of solution methods

The nature of a solution method for APP/DAPP problems indicates its essential properties from various aspects, which have been identified as *Scope*, *Mode*, and *Completeness* of the methods. The Mode of a solution method may be either *offline* or *online*, which

Table 4
Relevant works in APP/DAPP Solution Approach features.

SOLUTION APPROACH	**Graph-based**	**Blocking graph** [18,19,33,36,68,69,79,81] **And/or graph** [84] **Liaison graph** [50] **Connector-based graph** [89] **Visibility graph** [35,37,82] **Roadmap- and constraint-based** [66] **Contact state graph** [46]
	Grid-based	**Potential fields** [31,48,51–54] **Fast marching method** [55] **Maximum clearance** [45]
	Sampling-based	**Probabilistic Roadmaps (PRM)** [56,76,78] **Rapidly exploring random trees (RRT)** [13,32,39,40,58,59,61–64,70,75]
	Space decomposition	**Separating directions** [20,57,67,71,73,74,88,90] **Swept volume** [38,72,87] **Motion space** [17,65,77] **Cell decomposition** [49]
	Interactive	**Interactive path planning** [42,60,83,85] **Interactive path verification** [80,86] **Interactive path correction** [51–53]

is directly affected by the mode of the APP/DAPP problem and depends on whether or not (dis)assembly path planning can be completed before executing the plan. The Scope of a solution method is *global* if all the information in the environment is considered and used in path planning; whereas it is *local* if planning is based on limited and local information. In *mixed* scope, global and local information are integrated to plan in a known environment while responding to unexpected local changes. The Completeness aspect pertains to the ability and assurance of a method in finding a feasible (path planning) solution, or reporting nonexistence of such a solution. Solution methods may either be fully complete, have a weaker form of completeness, or not providing any guarantee to find a solution.

3.2.1. Mode

The mode of a (dis)assembly path planning method can be either offline or online. Offline path planners require a complete map of the assembly environment (including geometry and configurations of assembly parts, surrounding obstacles, tools, and the assembler) be known *a priori*. Since in many cases the status of the environment does not change during the (dis)assembly path planning process, a large number of planners have been developed for offline mode, including [13,17–19,31,33,35–39,54,56,58,59,63–83].

Online (dis)assembly path planning is required when a map of the assembly environment is not (fully) known, an unexpected obstacle is encountered during the execution of a preplanned path, or, if in addition to geometrical constraints, some other constraints (of mechanical or physical nature) are discovered during the assembly operations which call for replanning the paths of parts. Since online path planners run in real time and on computers that usually have very limited computing resources, they have to be efficient in terms of both memory utilization and time. For example, in [62] an online path planner uses a set of planners which are adaptively selected for more efficient path planning depending on the current assembly status.

3.2.2. Scope

The scope of a (dis)assembly path planner may be global, local or mixed. *Global* planners take into account the whole assembly environment, including all obstacles and start and goal configurations of parts, and can find global optimal solutions although they may spend high computational time and memory since they grow exponentially with the number of C-space dimensions. Because of using the whole information about the workspace and assembly, most Graph-based and Space Decomposition methods have global scope, including the Blocking Graph, Visibility Graph, Roadmap- and Constraint-based, Separating Directions, Swept Volume, and Motion Space methods. Interactive Path Verification methods [80,86] can also be classified as global planers since they employ the overall information about the parts and the assembly in planning (dis)assembly paths.

When the whole assembly environment is not known *a priori* or the assembly environment is dynamic, it is difficult for global methods to compute complete (dis)assembly paths in real time while avoiding collisions with obstacles and other parts. Instead, *local* or reactive solution methods are used, which are usually fast but do not guarantee to find a path, even if one exists. In planning a (dis)assembly path for a part, local path planners take into account only nearby obstacles and so find a path without a global knowledge about the environment. Generally, Grid-based APP/DAPP solution methods are considered local methods, as the Potential Fields, Fast Marching, and Maximum Clearance methods since they construct a path by repeatedly moving from a grid node or cell to its best local neighbor.

Methods with *mixed* global–local scope try to benefit from the advantages of both global and local methods. These methods usually use the global information to search for a rough obstacle-free region or capture the connectivity of the C-space by random sampling in the whole space, and then employ local planners to connect safe configurations and yield the final path. The PRM and RRT methods and most Interactive path planning methods [42,60,85] are among planners with mixed scope, as is the Cell Decomposition [49] method which uses an octree and A* search for finding a free channel, and then sampling random nodes and locally connecting them by the RRT method.

3.2.3. Completeness

Completeness is one of most important and desirable properties of any (dis)assembly path planner, and requires that the planner (1) find a solution if one exists, and (2) find out that no solution exists if that is the case. Completeness has different levels of strength, ranging from (fully) complete to incomplete.

Complete (dis)assembly methods are guaranteed to satisfy the above two conditions by building a discrete representation of the assembly workspace which exactly reflects the topology and connectivity of the original problem. All Graph-based methods (i.e., Blocking Graph, Visibility Graph, and Roadmap- and

Constraint-based), as well as three (out of four) Space Decomposition methods (i.e., Separating Directions, Swept Volume, and Motion Space) can be considered complete.

Two weaker forms of completeness are*Resolution-Completeness and Probabilistic Completeness*: Resolution-Complete methods can guarantee to find a solution if the accuracy needed for finding a solution (i.e. planning a path) is within a predefined resolution of discretizing the search space. Thus, if the only solution passes through a very narrow passage that the algorithm has not detected (covered by fine-enough cells), then it will (wrongly) report that no feasible path exists. On the other hand, in case that no predefined resolution threshold is set and no solution exists, then Resolution-Complete methods may continue searching with finer and finer decompositions until reaching infinitely small cells, and fail to report nonexistence of a solution. All Grid-based methods (i.e., Potential Fields, Fast Marching Method, and Maximum Clearance) are considered resolution-complete. The main drawback of resolution-complete methods is that most of the computational time is spent on collision checking. Another drawback is the system instability due to the probabilistic nature of random walks used in escaping from local minima, which leads to high variations in runtimes when finding a solution for a specific problem, and thus a low robustness of the method.

Correspondingly, *Probabilistic Complete* methods can guarantee to find a solution if they are provided with infinite time for solving the path planning problem; that is, the probability of finding a solution approaches 1 if the planning time approaches infinity. These methods heavily rely on stochasticity because instead of explicitly constructing the C-space, they try to map the continuous space into a graph of finite nodes and edges such that the connectedness of the C-space is captured and preserved. Although probabilistic complete methods do not guarantee to find a solution to complex problems within reasonable time, running them for several times may help with finding a feasible path or reporting that no solution is likely to exist. Obviously, Sampling-based methods (including PRM and RRT) are Probabilistic Complete.

By implementing *Probabilistically Resolution-Complete* methods, the probability of finding a free path converges toward 1 if there exists one at the resolution of the grid and when the running time grows toward infinity. An example of such method is presented in [31], which uses both a grid for path planning by the Potential Fields method, and a Randomized Path Planner for escaping from local minima. Also, in the Cell Decomposition method presented in [49], the octree decomposition is first used to find an obstacle-free channel (which is resolution-complete), and then the RDT method is used to find a path in the channel (which is probabilistic complete).

Finally, *Incomplete* methods usually employ greedy or heuristic methods which do not guarantee to find a solution at all since they do not search the C-space systematically, but rather at random. Such methods may get stuck in local minima forever if not equipped with effective procedure for escaping from them. Heuristic methods are mostly used when an exhaustive search of the solution space is impractical due to its huge size, and still may produce satisfactory results within reasonable runtime. For example, in [80] various metaheuristic methods have been implemented to generate assembly paths, which are then verified by a human operator through simulation. Also, in [83] assembly path planning for the parts of a decelerator is performed manually and interactively by a human operator, which is not guaranteed to find a solution for complex assemblies or realize the nonexistence of a feasible path.

The main aspects of nature of APP/DAPP solution methods, along with their respective relevant articles are shown in Table 5. For better visualization, the Euler–Venn diagram of this table is presented in Fig. 15.

4. Discussion

In this section some observations and analyses are presented related to the in-depth review and taxonomies of the previous sections, which deal with the frequency, applicability, and trends of solution approaches and their nature (mainly in terms of completeness), as well as industrial or synthesized applications for which APP/DAPP problems have been proposed, and finally some open problems and research areas in the field of (dis)assembly path planning.

4.1. Solution approaches and their nature

As mentioned earlier, APP/DAPP problems have been solved through five major categories of Graph-based, Grid-based, Sampling-based, Space Decomposition, and Interactive approaches. However, based on our review, among the implemented solution methods the four methods that collectively comprise more than half of the reviewed articles are RRT, Blocking Graph, Separating Directions and Potential Fields methods. Fig. 16 shows the number of published papers on different APP/DAPP solution methods.

It is noted that after extensive searches in numerous scientific databases, a total of 60 papers were found in the literature that tackled APP/DAPP problems in one way or another. We limited our search specifically to publications which explicitly dealt with APP/DAPP problems, and did not consider papers merely on ASP/DASP problems.

Graph-based methods are among the first methods used for APP/DAPP, and have been implemented most (16 out of 60). These methods are still used actively especially for the Motion Stability and Maintainability problems. Although applicable for parts with simple geometries, these methods have high computational requirements for complex parts. On the other hand, Space Decomposition methods can successfully handle (dis)assembly path planning of parts with complex shapes that can be (dis)assembled via simple single- or multi-step translational or translational-and-rotational movements. In these methods the dimension of the C-space is dependent on the complexity of the required movements.

The Sampling-based methods are the next most implemented methods after the Graph-based ones, probably because of their ability in solving high-dimensional or complex APP/DAPP problems (generally caused by large number of parts in assembly). Interestingly, the RRT method has been used more frequently than the PRM method; the reason is that unlike PRM which is a multi-query planner and takes random samples from the whole C-space, the RRT is a single-query random path planner that builds a tree specific to the given start and goal configurations, and besides, the search can be controlled and biased toward the goals of parts. In the case of APP, the goal of a part is its final configuration, whereas in the case of DAPP it is somewhere outside the bounding box of the assembly. Therefore, bidirectional RRT can be effectively used for APP problems, but not for DAPP problems where parts do not have specific goal locations. Despite their success, the 'narrow passage' problem remains as the main challenge for Sampling-based methods, and therefore coping with it more effectively is still at the focus of path planning researchers. Moreover, there are other sampling-based path planning methods (e.g. Expansive-Spaces Trees (EST) [109], Sampling-based Roadmap of Trees (SRT) [110], etc.) which have not found applications in APP/DAP problems.

Grid-based methods have been especially successful in solving the Maintainability problem, which seeks to find a collision-free path for disassembling a part out of a whole assembly. This is perhaps due to the fact that in extricating a single part from an assembly, other parts should not move and thus feasible solutions generally go through narrow passages. Grid-based methods effectively capture and preserve the connectivity of the search

Table 5
Relevant works in each aspect of nature of APP/DAPP solution methods.

NATURE	**Mode**	**Online** [32,40,42,45,48,49,51–53,61,62] **Offline** [13,17–20,31,33,35–39,46,50,54–60,63–90]
	Scope	**Global** [17–20,33,35–38,46,50,57,65–69,71–74,77,79–84,86–90] **Local** [31,45,48,51–55] **Mixed** [13,32,39,40,42,49,56,58–64,70,75,76,78,85]
	Completeness	**Complete** [17–20,33,35–38,42,46,50,57,65–69,71–74,77,79,81,82,84–90] **Resolution-Complete** [45,48,51–55] **Probabilistic Complete** [13,32,39,40,56,58–64,70,75,76,78] **Probabilistically Resolution-Complete** [31,49] **Incomplete** [80,83]

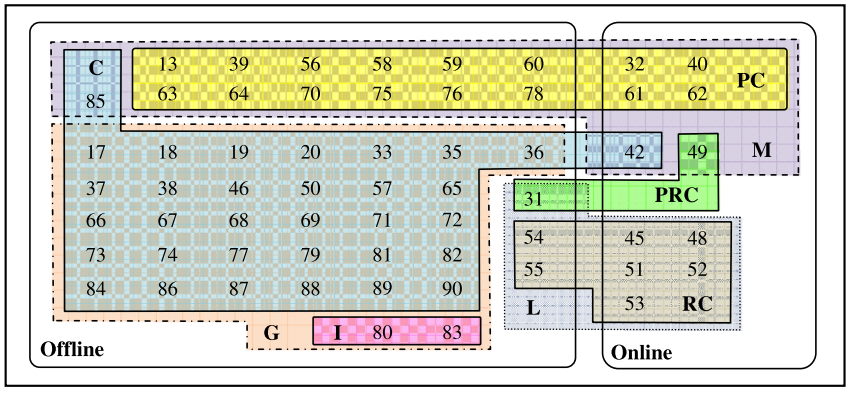

Fig. 15. Euler–Venn diagram of the Table 5. The numerals indicate reference numbers and the abbreviations stand for: C = complete, RC = resolution complete, PC = probabilistic complete, PRC = probabilistically resolution complete, I = incomplete, G = global, L = local, and M = Mixed (Global + Local).

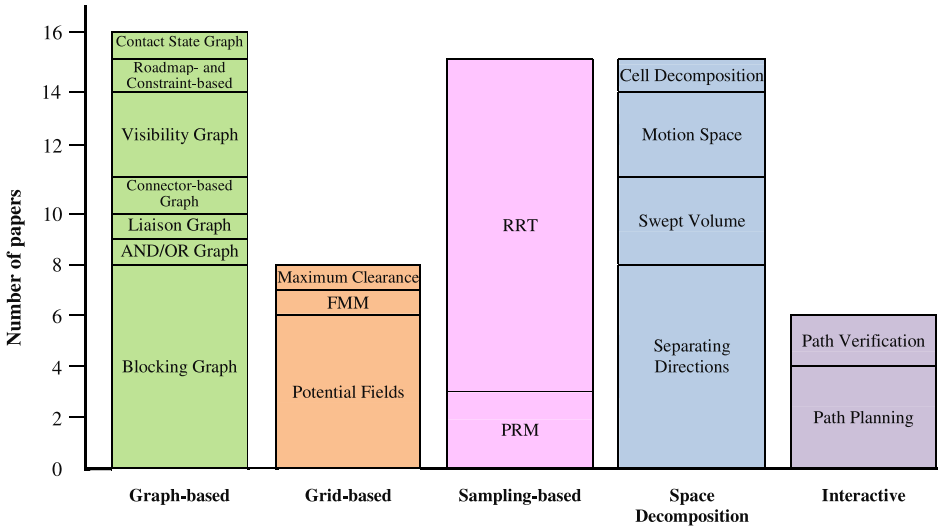

Fig. 16. Number of papers in major categories of APP/DAPP solution approaches in the reviewed 60 researches.

space and therefore plan (dis)assembly paths efficiently. Interactive methods which employ human intelligence through software packages or haptic devices are relatively the newest methods developed for APP/DAPP problems. Regarding the few number of publications using these methods (only 5–6), their recency (since 2012), and their somewhat different nature, comparing Interactive methods with other 'traditional' path planning methods was not possible.

Fig. 17 shows a chronological view of the reviewed papers in the five major categories. It is observed that the Graph-based, Grid-based, and Space Decomposition approaches were the main APP/DAPP planners before 1998. After that, however, Sampling-based methods were implemented increasingly and soon became the dominant approach especially in recent years, as they constitute 4 out of 15 works in just 2013. This diagram shows a trend of growing implementation of Sampling-based methods, which is expected to continue for the coming few years. Also, Interactive methods will supposedly find significance among other approaches thanks to the current advances in hardware (e.g., human–machine interfaces, haptic devices) and software (virtual reality, integrated and modular commercial packages).

Since (dis)assembly sequence planning (ASP/DASP) and (dis)assembly path planning (APP/DAPP) are two main components of the whole (dis)assembly planning (AP/DAP) problem, they are better to be treated at the same time. While some works perform ASP and APP sequentially or iteratively through some

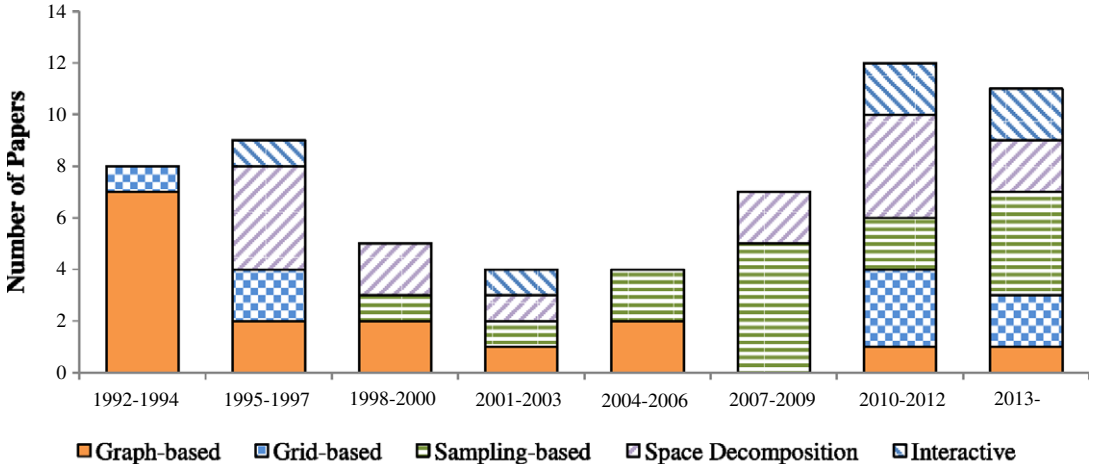

Fig. 17. Timeline of the frequency of works by major APP/DAPP approaches.

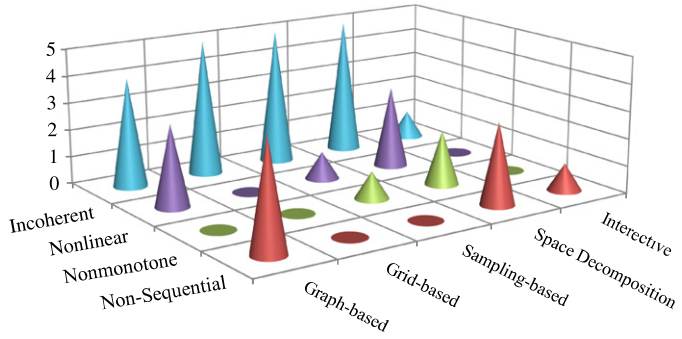

Fig. 18. Number of papers presented in various approaches according to the complexity of their considered APP/DAPP problem model.

backward–forward loops (as in [17,18,20,32,33,36–38,41,42,50,51,53,58,65,68,69,71–74,76,77,79,80,82,83,85–87,90,111]), a unified model and solution methods for solving these problems concurrently while considering their interactions and interdependence have yet to be developed, and thus can be regarded as an open problem.

Trying to determine the shares of solution methods according to their completeness, it was found out that complete methods (consisting all Graph-based and most Space Decomposition methods) have been used most frequently (56%) until 1997, although their computational complexity is prohibitive for complicated assemblies. On the other hand, weaker forms of completeness, i.e., resolution-completeness and probabilistic completeness, have been gradually introduced and applied in APP/DAPP since 1995 due to their less computational complexity (with 10% and 27% of all contributions, respectively). Finally, probabilistically resolution-complete and incomplete methods have been implemented less than the other three approaches, each with a 3.5% share.

Fig. 18 illustrates the number of papers presented in various approaches according to the complexity of their considered APP/DAPP problem model (also refer to Table 6). Of the reviewed 60 papers, only 27 papers dealt with APP/DAPP problem having one or two complexity challenges of Incoherency, Nonlinearity, Non-monotonicity, and Non-sequentiality (reaching to 38 individual items).

As the figure suggests, Sampling-based and Grid-based methods have not been used for non-sequential products, which require more than two-handed (dis)assembly. This is due to the increased complexity of simultaneous path planning for more than two parts. Non-monotone APP/DAPP problems which require considering intermediate placements for parts have been seldom solved by different approaches since it imposes various complexities. For instance, in Graph-based methods it leads to increased additional nodes on the search graph; in Grid-based methods it brings about preemptions in path planning as the paths of a part may be abandoned and restarted later, after path planning of another (blocking) part; or in Interactive methods the number of human interventions may increase.

The above complexities lead to increased runtimes which call for more effective and efficient solution methods. In order to handle nonlinearity, the planner must investigate all possible combinations of the assembly parts (amounting to 2^n for an n-part assembly) in order to determine which and how many subassemblies must be formed and entered into the main assembly as a compound unit component. That is why only those solution methods that do not suffer from large number of assembly components have been more successful in handling this issue. For example, Sampling-based methods are not too much sensitive to the dimensions of the C-space as they sample a limited number of random nodes in the search space, or in the Space Decomposition approach the search space dimension depends more on the complexity of allowed motions (e.g. refer to Section 3.1.4.3: Motion Space) rather than on the number of assembly components. Also, in the Graph-based approach, the parts, a group of parts, or connector-based assembly elements and their mutual relations (in terms of blocking, forces, constraints, etc.) are mapped into a graph (not in C-space) and appropriate paths are found using this graph. Finally, it is observed that almost all of the major approaches have been able to solve incoherent (dis)assembly path planning problems.

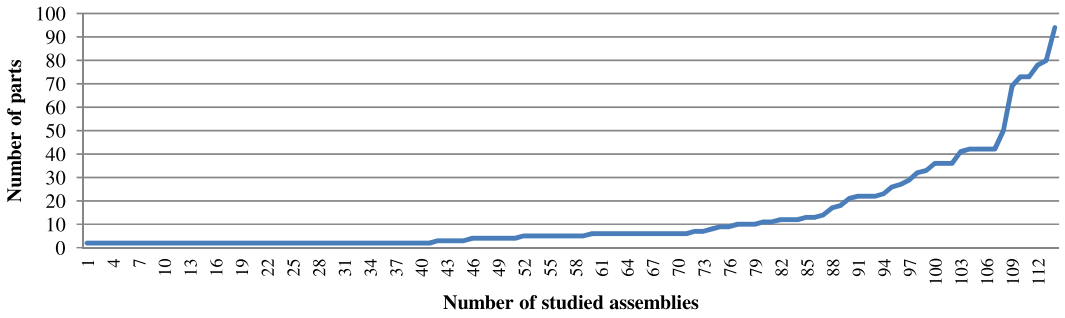

Fig. 19. Distribution of the number of parts in the studied assemblies of APP/DAPP-related papers.

4.2. Applications

Most of the reviewed APP/DAPP planners have been described, tested and run on various example assemblies, which we categorized into two sets of *Industrial* applications (representing real-world industrial products or their simplified models, except for two medical examples in [39,70]), and *Puzzle-like* assemblies (representing non-industrial, synthetic assemblies). Table 6 lists the reviewed 60 works, their adopted approach and method, reference number, completeness, and complexity level of the model of their solved problem(s). The Table also provides details of the applications each research has dealt with, as well as the number of parts each application had. Some works were applied on both industrial and puzzle-like assemblies, some works tackled specific applications for the first time (which are indicated in italics), and some provided no examples. The number of parts in assemblies ranged from 2 to 94, with an average of 13 parts per assembly. It is noted that we considered the assembly in [38] with 731 aircraft parts as outlier and so did not enter in the above mentioned range and average calculations. Fig. 19 illustrates the number of parts versus the number of studied examples, sorted in ascending order, which implies that 79 examples had less than or equal to 10 parts in assemblies, of which 41 had only 2 parts.

Unfortunately there are no benchmark puzzle-like or industrial assemblies to enable the APP/DAPP community to test and evaluate the performance of their (dis)assembly path planners. Such standard benchmark assemblies would be valuable tools for researchers in analyzing and validating their (dis)assembly path planners, and would allow reliable comparative studies and analyses. So developing such benchmarks could be the subject of future researches.

As shown in Table 6, among the studied 119 examples, 77 were actual or simulated industrial assemblies and 42 were puzzle-like examples, which shows the importance of the APP/DAPP problem in manufacturing industries.

Fig. 20 illustrates the number of examples provided in the reviewed papers classified into four main application groups of Automotive industry, Aircraft industry, Miscellaneous industrial devices, and Puzzle-like assemblies. The figure implies that more than half of the industrial examples were related to the automotive (29 instances) or aircraft (13 instances) industries.

Fig. 21 and Table 7 present the number and percentage of the main application groups in the examples solved by the major solution approaches. As the table suggests, the Sampling-based approach has been implemented for assemblies in automotive industry more than other approaches (with 33.96%) thanks to its computational efficiency in dealing with problems with numerous assembly parts (having C-spaces with high dimensions). Nevertheless, it has seldom been implemented for solving APP/DAPP problems in Aircraft industry (only 1.89%), which is probably due to the sensitivity of the matter, though a careful causal analysis is needed for finding out the reasons. It is worth to note that Interactive methods have been most applied in industrial examples (with a total of 83.3%), which shows that they have been most useful in real-world commercial applications relative to other approaches.

4.2.1. Commercial impact

Some commercial software solutions have been developed for efficient Computer Aided Assembly Planning (CAAP). For example, *KineoWorks*™was developed by Siemens PLM Software to automatically generate a collision-free (dis)assembly path for rigid parts inside digital mockups. This package can be integrated in widely used CAD/CAM software like *CATIA* and *DELMIA* (from Dassault Systèmes), as well as *NX*, *Vizmockup*, and *Process Simulate* (all from Siemens PLM Software). The *Voxmap Point-Shell (VPS)* system from the Boeing Company is a collision-detection library optimized for use with haptic interfaces and allows haptic devices to manipulate and move arbitrarily-shaped objects through a scene in order to plan (dis)assembly paths. Also, Sensible© PHANTOM™haptic device is used to add the sense of touch to the assembly path planning process of the Boeing and General Electric companies.

Additionally, some of the papers reviewed in this article used their path planners for real-world commercial assemblies, including:

- The automated system in [31] was encapsulated in *ProductVision*, a commercial software developed at General Electric Corporate R&D to provides a visual environment on UNIX workstations to generate assembly paths for an irregularly-shaped line replaceable unit (LRU) from a set of pipes.
- The disassembly path planner in [32] was used to complement the *REVIMA* Project (a VR tool aiming to create hardware and software tools for realistically simulating maintenance operations on aircraft engines and equipment). The project provides a haptic interface with a big workspace and realistic and interactive visualization of a virtual mock-up. The path planner provided additional analysis information such as precedence of removals that can be used in the interactive environment to ensure defining correct maintenance routines prior to construction of any physical mock-up.
- The approach presented in [42] was part of a project to develop a 'Virtual Design, Service, and Demanufacture Studio' in which by using automatically generated information and geometry of components, the designer decides on the component or fastener to be disassembled and the direction of removing them by interactively disassembling the prototype in the studio,
- The automatic path planner for wiring harness installation [45] was a part of the Virtual simulation and training of assembly and service processes in digital factories (the VISTRA EU-FP7-Project).

Table 6
Characteristics and applications of the reviewed researches in APP/DAPP.

Solution Approach		Reference	Example (and number of parts)	Application Type	Completeness	Complexity			
						Sequentiality	Monotonicity	Linearity	Coherence
GRAPH-BASED	Blocking Graph	[18]	*Removing parts of blocks assembly* (2, 4, 5)	puz	C	S	M	L	C
		[19]	*Removing toleranced parts from an engine* (6)	ind	C	NS	M	L	I
		[33]	Removing parts of a model aircraft combustion engine (42)	ind	C	S	M	L	C
		[36]	*Removing parts of a model aircraft engine* (13)	ind	C	S	M	L	C
		[68]	Removing parts of a crate (5)	ind	C	S	M	L	C
		[69]	N/A	N/A	C	S	M	L	C
		[79]	*Removing parts of cubes assembly* (4)	puz	C	S	M	L	C
			Removing parts of: *electric bell* (22), *model aircraft combustion engine* (42), transmission (21), friction testing (skin) machine (36) *a crate* (5)	ind					
		[81]	*Removing parts of a latch* (3)	puz	C	S	M	L	C
			Removing parts of: an electric bell (22), friction testing (skin) machine (36), a model of aircraft combustion engine (42)	ind					
	AND/OR Graph	[84]	*Removing parts of a simple assembly* (4)	puz	C	S	M	NL	I
	Liaison Graph	[50]	*Extracting the parts of a table-top vise assembly* (18)	ind	C	NS	M	NL	C
	Connector-based Graph	[89]	*Constructing the connector-based graph for a multi-plate clutch* (27)	ind	C	NS	M	NL	C
	Visibility Graph	[35]	*Moving a part through map of parts* (6)	puz	C	S	M	L	I
		[37]	*Extracting a box from its associated cover* (2)	puz	C	S	M	L	C
		[82]	*Removing parts of a model aircraft vertical tail* (N/A)	ind	C	S	M	L	C
	Contact State Graph	[46]	*Inserting an elastic tube into a rigid groove* (2)	ind	C	S	M	L	C
	Roadmap- and Constraint-based	[66]	*Removing a bolt from a pump* (2)	ind	C	NS	M	L	I
GRID-BASED	Potential Fields	[31]	*Removing an irregularly-shaped line replaceable unit (LRU) from a set of pipes* (2), *inserting a part into a twisted cavity without deformation* (2)	ind	PRC	S	M	L	I
		[48]	*Extracting gear, shaft and lid from a Box* (3)	ind	RC	S	M	L	C
		[51]	Removing parts of a 2D puzzle (6)	puz	RC	S	M	L	C
		[52]	*Removing parts of an aircraft model* (6)	ind	RC	S	M	L	I
		[54]	*Moving an LRU through obstacles while avoiding a heat source* (4)	ind	RC	S	M	L	I
		[53]	*Removing the parts of an aircraft model* (6)	ind	RC	S	M	L	I
	Fast Marching Method	[55]	*Inserting a peg in a hole* (2)	puz	RC	S	M	L	C
	Maximum Clearance	[45]	*Connecting a wiring harness to the engine compartment of a car* (2)	ind	RC	S	M	L	I

(continued on next page)

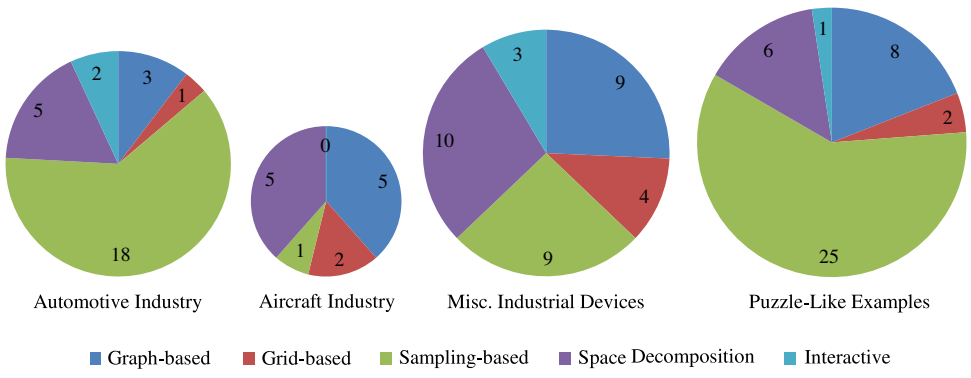

Fig. 20. Breakdown of the four main application groups by solution approaches in the reviewed 60 papers. The sizes of circles are proportional to the number of examples in each group.

4.3. Open problems

In addition to some open problems mentioned in Section 4.1 regarding APP/DAPP solution methods, in this section we put forth some other open problems and matters of future research mainly related to the model and nature of APP/DAPP problems.

4.3.1. Problem components

In an APP/DAPP problem, taking into account assembly parts, tools and the assembler altogether and simultaneously does reflect real-world conditions and problems more realistically, while at the same time increases the complexity of the problem. Due to the difficulty of this matter, as well as challenges in scheduling, path planning, and grasp/regrasp planning of available assembly tools and robots, the complete set of components have not been simultaneously considered in any APP/DAPP model. Therefore, this is an open issue in the field of assembly path planning, which can help in solving more sophisticated, real-world complex problems.

4.3.2. Problem constraints

Most APP/DAPP planners considered only geometric constraints of the assembly parts, whereas few articles took into account phys-

Table 6 (*continued*)

Solution Approach		Reference	Example (and number of parts)	Application Type	Completeness	Complexity			
						Sequentiality	Monotonicity	Linearity	Coherence
Sampling-based	Probabilistic Roadmaps (PRM)	[56]	*Assembling a driving unit into a car body* (2)	ind	PC	S	M	L	C
		[76]	*Removing parts of: a planar puzzle (plates)* (6), *a planar puzzle (box)* (6), *3D puzzles* (10), *Pentomino puzzles* (12)	puz	PC	S	M	L	C
		[78]	*Moving an angle bracket through a wall* (2)	puz	PC	S	M	L	I
			extracting a ring from a cam shaft (2), *extracting a power socket from a top hat rail* (2)	ind					
	Rapidly Exploring Random Trees (RRT)	[13]	*Removing: a pipe from a machinery room* (2), a wiper motor (2), a seat from the car body (2)	ind	PC	S	M	L	I
			Removing parts of an Alpha puzzle (2)	puz					
		[32]	*Extracting selected elements from a box* (9)	puz	PC	S	M	L	C
			Removing parts of: valve (11), *torque converter* (36), *admission of a car* (69), *differential* (78), *Extracting: a gear from its casing* (2), *the fixing key of the air filter of an engine* (2)	ind					
		[39]	*Extracting ligand from its associated protein* (2)[*med]	ind	PC	S	M	L	C
		[40]	Removing: *Radiator of a car* (2), a wiper motor (2), a seat from the car body (2)	ind	PC	S	M	L	C
		[58]	Removing parts of: planar (plates) (6), Pentomino (12), *2D Narrow* (3), *3D Narrow* (3) diagonal star puzzles (6)	puz	PC	S	NM	L	I
		[59]	*Removing parts of a five-part puzzle* (5)	puz	PC	S	M	NL	C
			Removing parts of: car chassis (23, 73), *crankshaft* (41), *radial crankshaft* (50)	ind					
		[61]	Extracting: a gear from its casing (2), the fixing key of air filter of an engine (2)	ind	PC	S	M	L	C
			Extracting: a cube from a complex volume in a labyrinth (2) *a cube through an S-shaped tunnel* (2)	puz					
		[62]	*Moving a block through sandwich wall and stairs blocks* (4), *moving a stick through parallel walls* (7), *Alpha puzzle* (2)	puz	PC	S	M	L	I
			Disassembling a flange and elbow (2)	ind					
		[63]	*Moving in an environment with obstacle and danger zones* (5)	puz	PC	S	M	L	C
		[64]	*Removing: a wiper motor from the car body* (2), *a seat from the car body* (2)	ind	PC	S	M	L	C
		[70]	*Removing parts of: a pump* (4), *a catheter assembly* [*med]	ind	PC	S	M	L	C
		[75]	*Removing a part from an aircraft hydraulic system* (2)	ind	PC	S	M	L	I
			Pulling a cylinder out of a mating hole (2), moving through *X-shape and XX-shape tunnels* (2), a *simple block* (2), *an L-shaped Beam going through a small window on the wall* (2), *a block going through one of the narrow passages in the obstacle* (2)	puz					
Space Decomposition	Separating Directions	[67]	Removing parts of a model aircraft combustion engine (12)	ind	C	NS	M	NL	C
		[71]	*Extracting: a cube from a table* (2), *a cube from a tetrahedron* (2)	puz	C	S	NM	NL	C
			Extracting a spacer from a side-plate (2)	ind					
		[73]	*Removing parts of: a tilted blender* (7), *a Discriminator* (11), a crate (5)	ind	C	S	M	L	C
		[74]	Removing parts of: a model aircraft combustion engine (42) an electric bell (22)	ind	C	S	M	NL	I
		[57]	*Removing the parts of a reduction gearbox* (9)	ind	C	S	M	L	C
		[88]	*Generating the exploded view of a cylinder gear reducer* (94)	ind	C	S	M	L	C
		[20]	*Removing the parts of: a simplified car model* (8), *a lifting device* (5), *a flange coupling* (6) and a heel pad clamp (14)	ind	C	NS	NM	L	I
		[90]	*Removing parts of a Washer* (20)	ind	C	S	M	L	C
	Swept Volume	[38]	*Removing parts of a component of the center wing of an aircraft* (731)	ind	C	S	M	L	I
		[72]	*Removing parts of fore flap of an aircraft* (80)	ind	C	S	M	L	I
		[87]	*Removing parts of an automotive generator* (17)	ind	C	S	M	L	C
	Motion space	[17]	N/A	N/A	C	S	M	L	C
		[65]	*Removing parts of a diagonal star puzzle* (6)	puz	C	NS	M	L	C
		[77]	*Removing parts of: an aircraft model* (32), *a headlight* (33), *a car chassis* (73)	ind	C	S	M	L	C
			Removing parts of a cube (10)	puz					
	Cell Decomposition	[49]	*Moving through: a scene with large free-space area* (10) *complex CAD models with narrow passages*	puz	PRC	S	M	L	I
Interactive	Interactive Path Planning	[83]	*Removing the parts of a mechanical decelerator* (6)	ind	IC	NS	M	L	C
		[42]	*Removing the parts of a center console assembly of a car* (5)	ind	C	S	M	L	C
		[85]	*Removing the parts of an active bevel gear assembly* (13)	ind	C	S	M	L	C
		[60]	*Passing an L-shape part through a narrow passage* (2)	puz	PC	S	M	L	I
	Interactive Path Verification	[80]	*Removing the parts of a worm gear reducer* (26)	ind	IC	S	M	L	C
		[86]	*Digital modeling of an engine turbine rotor* (29)	ind	C	S	M	L	C

Legend: C = complete, RC = resolution complete, PC = probabilistic complete, PRC = probabilistically resolution complete, IC = incomplete, puz = puzzle-like, ind = industrial application, S = sequential, NS = non-sequential, M = monotone, NM = non-monotone, L = linear, NL = nonlinear, C = coherent, I = incoherent, N/A = not available, [*med] = medical application. *Examples in italics indicate their appearance for the first time in the literature*.

Table 7
Percentages of various application groups per solution methods for 118 examples in the reviewed 60 papers.

Application group	Graph-based	Grid-based	Sampling-based	Space decomposition	Interactive
Automotive industry	12.00%	11.11%	33.96%	19.23%	33.33%
Aircraft industry	20.00%	22.22%	1.89%	19.23%	0.00%
Misc. industrial devices	36.00%	44.44%	16.98%	38.46%	50.00%
Puzzle-like examples	32.00%	22.22%	47.17%	23.08%	16.67%

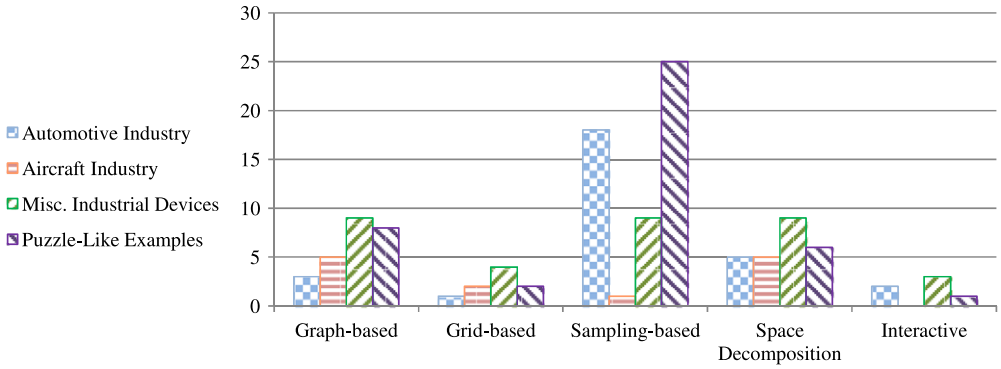

Fig. 21. Number of various application groups per solution methods for 119 examples in the reviewed 60 papers.

ical constraints like gravity and friction forces for (dis)assembly path planning or (dis)assembly stability problems. However, simultaneously applying geometrical, physical, and mechanical constraints in APP/DAPP models (which is the case in real-world applications) is still an open problem. This gains more importance especially in APP/DAPP problems with deformable parts, or when the scale of the problem is tightly fine. In such situations, considering merely geometrical constraints may lead to infeasible or unrealistic (dis)assembly paths so that the planner may not be able to achieve an existing path due to excessive simplifications in the model constraints.

4.3.3. Problem objective functions

Nearly all of the reviewed papers have sought to optimize a single particular objective function (as noted in Table 1), while in cases that optimality of multiple objective functions is considered or when different goals are in conflict with each other (e.g. when the goal is to simultaneously reduce path length and runtime), using multi-objective optimization techniques is more appropriate and sometimes necessary. Therefore, the objective functions of APP/DAPP problems can be further enriched in future contributions.

4.3.4. Problem complexity

As mentioned in Section 4.1, all the studied papers have tried to resolve only one or two challenging issues related to the complexity the problem, namely, non-sequentiality, non-monotonicity, nonlinearity, and incoherency. While complex real-world industrial assemblies are at the same time non-sequential, non-monotone, nonlinear, and incoherent, no research has simultaneously addressed all these issues so far, despite the fact that such complexities have been known and attempted to be overcome since 1992. Therefore, developing a powerful APP/DAPP planner in this respect is a future challenge.

4.3.5. Parts geometry

Aside from the article [19] published in 1997, tolerances in the geometry of assembly parts have not been considered in other articles. Such tolerances occur in real-world applications due to ripples and unevenness in surfaces of parts, measurement and detection errors of robot sensors, or errors in placing the parts in right positions, and thus incorporating part tolerances in APP/DAPP models leads to more realistic and successful (dis)assembly paths.

Another way of simplifying APP/DAPP models and reducing the C-space dimensions of flexible-body parts is to model it as an articulated object, which has only been considered in [39] with two fixed and mobile articulated objects. Yet, (dis)assembly path planning of products with more articulated parts with complex geometries is an open issue.

Then again, automatic assembly path planning for flexible objects is widely accepted as a very difficult problem in general. Because of the intrinsic characteristics of flexible parts like high elasticity, large deformations caused by external forces and moments, as well as complicated dynamic behavior, incorporating such parts in the (dis)assembly process increases the complexity of APP/DAPP problems significantly. Although in many cases the forms of parts do not change during the assembly process, there are still many assemblies that are composed of both rigid and flexible parts (such as ships, aircrafts and automobiles) which require considering deformations of flexible parts for automatic generation of (dis)assembly paths. As a result, simultaneous consideration and combination of flexible and rigid parts in an assembly path planner is still an open research issue.

5. Conclusions

Assembly and Disassembly Planning are two fundamental problems in any manufacturing industry, which in addition to their significant importance and impact on manufacturing and assembling operations, give better insight to part and product designers in creating more cost-effective products in manufacturability and maintainability respects. Assembly and Disassembly Planning problems encompass three main subproblems: ASP/DASP, ALB/DALB, and APP/DAPP. While survey papers exist for the first two subproblems, i.e., (dis)assembly sequence planning and (dis)assembly line balancing, no taxonomy or comprehensive review has been presented so far for the (dis)assembly path planning problem. As a result of an in-depth review, this paper attempts to provide a comprehensive review of the state of the art of APP/DAPP literature and proposes two taxonomies for problems and solution methods of this evolving field. The first taxonomy pinpoints the main properties of APP/DAPP problem models (like types and dimensions of components, geometry and movements of parts, constraints and objective functions, etc.), APP/DAPP problem nature (such as scale and various complexities), APP/DAPP solution approaches (consisting Graph-based, Grid-based, Sampling-based, Space Decomposition, and Interactive), and APP/DAPP solution nature (including Scope, Mode, and Completeness of methods). Also, some observations and analyses are presented on the frequency, applicability, completeness, and trends of the existing solution approaches and their nature, as well as industrial or synthesized applications solved by them. Finally, some open problems and research areas in the APP/DAPP field are highlighted.

The taxonomies incorporate and provide essential keywords and their hierarchical structures in all important concepts within the APP/DAPP discipline and provide sufficient knowledge and guidance for understanding and directly finding the right and exact information about researches on each aspect of APP/DAPP problems and solution methods. Through using the comprehensive frameworks of the taxonomies and the discussions provided, researchers can identify research gaps in APP/DAPP problems formulation and solution and thereby initiate novel researches in this field.

Appendix. Description of used acronyms

Acronym	Description	Acronym	Description
AABB	Axis-Aligned Bounding Box	IRM	Iterative Rotational Method
ACA	Ariadne's Clew Algorithm	ISM	Iterative Spread Method
ACO	Ant Colony Optimization	ITM	Iterative Translational Method
AIS	Artificial Immune Systems	LG	Linguistic Geometry
ALB	Assembly Line Balancing	LP	Linear Programming
ANN	Artificial Neural Network	LRU	Line Replaceable Unit
AP	Assembly Planning	MAD	Maintenance Assembly/Disassembly
APP	Assembly Path Planning	ML-RRT	Manhattan-like RRT
ASP	Assembly Sequence Planning	MRP	Material Requirement Planning
B-Rep	Boundary Representation	NDBG	Non-Directional Blocking Graph
CAAP	Computer Aided Assembly Planning	NLP	Non-Linear Programming
CT	Control Theory	OBB	Object-aligned Bounding Box
DALB	Disassembly Line Balancing	OGRRT	Obstacle and Greedy rule-based RRT
DAP	Disassembly Planning	PF	Potential Fields
DAPP	Disassembly Path Planning	PP	Path Population
DASP	Disassembly Sequence Planning	PRM	Probabilistic Roadmaps
DBG	Directional Blocking Graph	PSO	Particle Swarm Optimization
DFG	Directional Force Graph	RDT	Rapidly-exploring Deterministic Tree
DFMA	Design For Manufacture and Assembly	RMM	Recursive Midpoint Method
DP	Dynamic Programming	RMP	Robot Motion Planning
DRRT	Danger-zones RRT	RPP	Randomized Path Planner
DWHG	Disassembly Weighted Hybrid Graph	RRT	Rapidly-exploring Random Trees
EDR	Error Detection and Recovery	RWM	Random Walk Method
EIM	Extended Interference Matrix	SA	Simulated Annealing
ES	Expert System	SALBP	Simple Assembly Line Balancing Problem
EST	Expansive-Spaces Trees	SI	Stability Index
FC	Fuzzy Control	SP	Sequence Population
FIM	First Intersection Method	SPI	Structural Preference Index
FL	Fuzzy Logic	SRT	Sampling-based Roadmap of Trees
FMM	Fast Marching Method	T-RRT	Targetless RRT
GA	Genetic Algorithms	TS	Tabu Search
GAAA	Genetic Algorithm and Ant colony Algorithm	VG	Visibility Graph
GALBP	Generalized Assembly Line Balancing Problem	VM	Virtual Manufacturing
GT	Game Theory	VP	Virtual Prototyping
GVD	Generalized Voronoi Diagram	VPS	Voxmap Point-Shell
I-ML-RRT	Iterated Manhattan-like RRT	VR	Virtual Reality
IP	Integer Programming		

References

[1] Pan C. Integrating CAD files and automatic assembly sequence planning (Doctoral dissertation), Iowa State University; 2005.
[2] Kavraki L, Latombe J-C, Wilson RH. On the complexity of assembly partitioning. Inform. Process. Lett. 1993;48:229–35.
[3] Jiménez P. Survey on assembly sequencing: a combinatorial and geometrical perspective. J Intell Manuf 2013;24:235–50.
[4] Lv H, Lu C. An assembly sequence planning approach with a discrete particle swarm optimization algorithm. Int J Adv Manuf Technol 2010;50:761–70.
[5] Marian RM. Optimisation of assembly sequences using genetic algorithms (Doctoral dissertation), Adelaide, Australia: University of South Australia; 2003.
[6] Scholl A, Becker C. State-of-the-art exactand heuristic solution procedures for simple assembly line balancing. European J. Oper. Res. 2006;168:666–93.
[7] Capacho Betancourt L. Asalbp: The alternative subgraphs assembly line balancing problem. formalization and resolution procedures (Doctoral dissertation), Technical University of Catalonia; 2007.
[8] Tasan SÖ, Tunali S. Improving the genetic algorithms performance in simple assembly line balancing. In: Computational science and its applications (ICCSA). Springer; 2006. p. 78–87.
[9] Rashid MFF, Hutabarat W, Tiwari A. A review on assembly sequence planning and assembly line balancing optimisation using soft computing approaches. Int J Adv Manuf Technol 2012;59:335–49.
[10] Reif JH. Complexity of the mover's problem and generalizations extended abstract. In: Proceedings of The 20th annual IEEE conference on foundations of computer science, 1979, p. 421–7.
[11] Canny J. The complexity of robot motion planning. MIT Press; 1988.
[12] Schwartz JT, Sharir M. On the piano movers problem. ii. general techniques for computing topological properties of real algebraic manifolds. Adv Appl Math 1983;4:298–351.
[13] Zhang L, Huang X, Kim Y, Manocha D. D-Plan: Efficient collision-free path computation for part removal and disassembly. J Comput-Aided Des Appl 2008;5:774–86.
[14] Latombe J-C. Robot motion planning. Springer; 1991.
[15] LaValle SM. Planning algorithms. Cambridge University Press; 2006.
[16] Lozano-Perez T. Spatial planning: A configuration space approach. IEEE Trans. Comput. 1983;100:108–20.
[17] Halperin D, Latombe J-C, Wilson RH. A general framework for assembly planning: The motion space approach. Algorithmica 2000;26:577–601.
[18] Rakshit S, Akella S. The influence of motion path and assembly sequence on the stability of assemblies. Robot Sci Syst 2013;IX.
[19] Latombe J-C, Wilson RH, Gazals F. Assembly sequencing with toleranced parts. Comput Aided Des 1997;29:159–74.
[20] Srinivasan H, Shyamsundar N, Gadh R. A framework for virtual disassembly analysis. J Intell Manuf 1997;8:277–95.
[21] Lambert AJ. Disassembly sequencing: A survey. Int J Prod Res 2003;41:3721–59.
[22] Scholl A. Balancing and sequencing of assembly lines. Physica-Verlag; 1999.
[23] Blum M, Griffith A, Neumann B. A stability test for configurations of blocks, Artificial Intelligence Memo No. 188, Massachusetts Institute of Technology, 1970.
[24] Boneschanscher N, Van Der Drift H, Buckley SJ, Taylor RH. Subassembly stability. In: Aaai, 1988, p. 780–5.
[25] Palmer RS. Computational complexity of motion and stability of polygons. In: Cornell University, 1989.
[26] Wolter JD, Trinkle JC. Automatic selection of fixture points for frictionless assemblies. In: Proceedings of IEEE international conference on robotics and automation, 1994, p. 528–34.
[27] Mattikalli R, Baraff D, Khosla P. Finding all stable orientations of assemblies with friction. IEEE Trans. Robot. Autom. 1996;12:290–301.
[28] Mattikalli R, Baraff D, Khosla P, Repetto B. Gravitational stability of frictionless assemblies. IEEE Trans. Robot. Autom. 1995;11:374–88.
[29] Mosemann H, Rohrdanz F, Wahl FM. Stability analysis of assemblies considering friction. IEEE Trans. Robot. Autom. 1997;13:805–13.
[30] Mosemann H, Rohrdanz F, Wahl F. Assembly stability as a constraint for assembly sequence planning. In: Proceedings of IEEE international conference on robotics and automation, 1998, p. 233–8.
[31] Chang H, Li T-Y. Assembly maintainability study with motion planning. In: Proceedings of IEEE international conference on robotics and automation, 1995, p. 1012–9.
[32] Aguinaga I, Borro D, Matey L. Path-planning techniques for the simulation of disassembly tasks. Assem Autom 2007;27:207–14.
[33] Wilson RH, Kavraki L, Latombe J-C, Lozano-Pérez T. Two-handed assembly sequencing. Int J Robot Res 1995;14:335–50.
[34] Kavraki LE, Kolountzakis MN. Partitioning a planar assembly into two connected parts is NP-complete. Inform. Process. Lett. 1995;55:159–65.
[35] Haicheng L, Yuan L, Jianfeng Y, Yuan Z. Path planning algorithm for assembly of complex product based on V-map and ant colony optimization algorithm. In: IEEE 3rd international conference on advanced computer theory and engineering, ICACTE, 2010, p. V5-398-V395-402.
[36] Lee S, Moradi H. Disassembly sequencing and assembly sequence verification using force flow networks. In: Proceedings of IEEE international conference on robotics and automation, 1999, p. 2762–7.
[37] Oliver JH, Huang H-T. Automated path planning for integrated assembly design. Comput Aided Des 1994;26:658–66.
[38] Hui C, Yuan L, Kai-Fu Z. Efficient method of assembly sequence planning based on GAAA and optimizing by assembly path feedback for complex product. Int J Adv Manuf Technol 2009;42:1187–204.
[39] Cortés J, Jaillet L, Siméon T. Disassembly path planning for complex articulated objects. IEEE Trans Robot 2008;24:475–81.
[40] Ferré E, Laumond J-P, Arechavaleta G, Estevès C. Progresses in assembly path planning, In: Int. conf. on product lifecycle management (Plm'05), 2005, p. 373–82.
[41] Givehchi M, Ng A, Wang L. Evolutionary optimization of robotic assembly operation sequencing with collision-free paths. J Manuf Syst 2011;30:196–203.
[42] Siddique Z, Rosen DW. A virtual prototyping approach to product disassembly reasoning. Comput Aided Des 1997;29:847–60.
[43] Coutee AS. Virtual assembly and disassembly analysis: An exploration into virtual object interactions and haptic feedback, 2004.
[44] Lim T, Ritchie J, Sung R, Kosmadoudi Z, Liu Y, Thin A. Haptic virtual reality assembly–moving towards real engineering applications. In: Advances in haptics. Intech; 2010. p. 693–722.
[45] Hermansson T, Bohlin R, Carlson JS, Söderberg R. Automatic assembly path planning for wiring harness installations. J Manuf Syst 2013.
[46] Luo Q, Xiao J. Haptic rendering involving an elastic tube for assembly simulations, In: The 6th IEEE international symposium on assembly and task planning: from nano to macro assembly and manufacturing, ISATP 2005, p. 53–9.
[47] Natarajan BK. On planning assemblies. In: Proceedings of the fourth annual symposium on computational geometry. ACM; 1988. p. 299–308.
[48] Gottschlich SN, Kak AC. AMP-CAD: Automatic assembly motion planning using cad models of parts. Robot Auton Syst 1994;13:245–89.
[49] Ladeveze N, Fourquet J-Y, Puel B. Interactive path planning for haptic assistance in assembly tasks. Comput Graph 2010;34:17–25.
[50] Lee S, Shin YG. Assembly coplanner: Co-operative assembly planner based on subassembly extraction. J Intell Manuf 1993;4:183–98.
[51] Yoon J. Assembly simulations in virtual environments with optimized haptic path and sequence. Robot Comput-Integr Manuf 2011;27:306–17.
[52] Hassan S, Yoon J. Haptic guided optimized aircraft maintenance assembly disassembly path planning scheme, In: IEEE international conference on control automation and systems (ICCAS), 2010, p. 1667–72.
[53] Hassan S, Yoon J. Haptic based optimized path planning approach to virtual maintenance assembly/disassembly (Mad). In: 2010 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE; 2010. p. 1310–5.

[54] Li T-Y, Chang H. Design for maintenance by constrained motion planning, In: Proceedings of 1995 symposium on computer integrated concurrent design, ASME design technical conference, 1995.
[55] Lee D, Kim MY, Cho H. Path planning for micro part assembly by using active stereo vision with a rotational mirror. Sensors Actuators A 2013;193:201–12.
[56] Carlson JS, Spensieri D, Söderberg R, Bohlin R, Lindkvist L. Non-nominal path planning for robust robotic assembly. J Manuf Syst 2013;32:429–35.
[57] Gao QF, Shang LL, Zhang LX. An efficient method of assembly path planning in virtual manufacturing system. Adv Mater Res 2012;468:550–8.
[58] Le DT, Cortés J, Siméon T. A path planning approach to (dis) assembly sequencing. In: IEEE International conference on automation science and engineering, case 2009, p. 286–91.
[59] Morato C, Kaipa K, Gupta SK. Improving assembly precedence constraint generation by utilizing motion planning and part interaction clusters. Comput Aided Des 2013.
[60] Yan Y, Poirson E, Bennis F. Integrating user to minimize assembly path planning time in PLM. In: Product lifecycle management for society. Springer; 2013. p. 471–80.
[61] Aguinaga I, Borro D, Matey L. Parallel RRT-based path planning for selective disassembly planning. Int J Adv Manuf Technol 2008;36:1221–33.
[62] Vallejo DR, Jones C, Amato NM. An adaptive framework for 'single shot' motion planning. In: Proceedings of IEEE/RSJ international conference on intelligent robots and systems (IROS) 2000, p. 1722–7.
[63] Fei P, Yao Z. Random triangle sampling path planning of assembly/disassembly in environment with dangerzones. In: IEEE international conference on measuring technology and mechatronics automation (ICMTMA), 2010, p. 972–6.
[64] Ferre E, Laumond J-P. An iterative diffusion algorithm for part disassembly. In: Proceedings of IEEE international conference on robotics and automation (ICRA) 2004, p. 3149–54.
[65] Fogel E, Halperin D. Polyhedral assembly partitioning with infinite translations or the importance of being exact. In: Algorithmic foundation of robotics VIII. Springer; 2009. p. 417–32.
[66] Garber M, Lin MC. Constraint-based motion planning using voronoi diagrams, In: Algorithmic foundations of robotics V, 2004, p. 541–58.
[67] Guibas LJ, Halperin D, Hirukawa H, Latombe J-C, Wilson RH. A simple and efficient procedure for polyhedral assembly partitioning under infinitesimal motions. In: Proceedings of IEEE international conference on robotics and automation, 1995, p. 2553–60.
[68] Lozano-Perez T, Wilson RH. Assembly sequencing for arbitrary motions. In: Proceedings of IEEE international conference on robotics and automation, 1993, p. 527–32.
[69] Lu T, Zhang B, Jia P. Assembly sequence planning based on graph reduction, In: Tencon' 93 proceedings of IEEE region 10 conference on computer, communication, control and power engineering, 1993, p. 119–22.
[70] Liu M, Liu J, He Y, Shang W. Research on assembly path planning and optimization of complex structures. Chin J Mech Eng 2013;49:97–105.
[71] Mosemann H, Bierwirth T, Wahl F, Stoeter S. Generating polyhedral convex cones from contact graphs for the identification of assembly process states. In: Proceedings of IEEE international conference on robotics and automation, ICRA'00, 2000, p. 744–9.
[72] Hui QXWSC. Space swept algorithm based assembly path planning method for aircraft. J Beijing Univ Aeronaut Astronaut 2010;6:012.
[73] Romney B. Atlas: An automatic assembly sequencing and fixturing system. In: Geometric modeling: theory and practice. Springer; 1997. p. 397–415.
[74] Romney B, Godard C, Goldwasser M, Ramkumar G. An efficient system for geometric assembly sequence generation and evaluation. Comput Eng 1995; 699–712.
[75] Shang W, Liu J, Ning R, Liu M. Computational path planner for product assembly in complex environments. Chin J Mech Eng 2013;26:282–92.
[76] Sundaram S, Remmler I, Amato NM. Disassembly sequencing using a motion planning approach. In: Proceedings of IEEE international conference on robotics and automation, ICRA 2001, p. 1475–80.
[77] Thomas U, Barrenscheen M, Wahl FM. Efficient assembly sequence planning using stereographical projections of C-space obstacles, In: Proceedings of IEEE international symposium on assembly and task planning, 2003, p. 96–102.
[78] Thomas U, Iser R. A new probabilistic path planning algorithm for (dis)assembly tasks. In: 41st international symposium on robotics (ISR) and 6th German conference on robotics (ROBOTIK). VDE; 2010. p. 1–6.
[79] Wilson RH. On geometric assembly planning (Doctoral dissertation), Stanford University: Computer Science Department; 1992.
[80] Da Xu L, Wang C, Bi Z, Yu J. AUTOASSEM: An automated assembly planning system for complex products. IEEE Trans Ind Inform 2012;8:669–78.
[81] Wilson RH, Latombe J-C. Geometric reasoning about mechanical assembly. Artificial Intelligence 1994;71:371–96.
[82] Yi S, Jianfeng Y, Yuan L, Haicheng Y. An assembly path planning algorithm for improving aircraft assembly. J Northwest Polytech Univ 2001;19:121–4.
[83] Jin X, Zhang T, Yang H. An analysis of the assembly path planning of decelerator based on virtual technology. Phys Procedia 2012;25:170–5.
[84] Homem De Mello LS, Sanderson AC. AND/OR graph representation of assembly plans. IEEE Trans. Robot. Autom. 1990;6:188–99.
[85] Hu D, Hu Y, Li C. Mechanical product disassembly sequence and path planning based on knowledge and geometric reasoning. Int J Adv Manuf Technol 2002; 19:688–96.
[86] Xu L, Wang C, Bi Z, Yu J. Object-oriented templates for automated assembly planning of complex products. IEEE Trans Autom Sci Eng 2013.
[87] Popescu D, Iacob R. Disassembly method based on connection interface and mobility operator concepts. Int J Adv Manuf Technol 2013;69:1511–25.
[88] Yu J, Xu L, Bi Z, Wang C. Extended interference matrices for exploded view of assembly planning. IEEE Trans Autom Sci Eng 2014;11:279–86.
[89] Tseng H-E, Li R-K. A novel means of generating assembly sequences using the connector concept. J Intell Manuf 1999;10:423–35.
[90] Zhang XF, Zhang SY, Qiu LM, An XH, Sa RN. Virtual disassembly simulation of varying target component for complex products. Adv Mater Res 2011;189: 2467–71.
[91] Wolter JD. On the automatic generation of plans for mechanical assembly (Doctoral dissertation), University of Michigan; 1988.
[92] Choset HM. Principles of robot motion: theory, algorithms, and implementation. MIT Press; 2005.
[93] Cottle R, Pang J, Stone R. The linear complementarity problem. New York: Academic Press; 1992.
[94] Nilsson NJ. A mobile automaton: an application of artificial intelligence techniques. In, Proceedings of the 1st international joint conference on artificial intelligence, Washington D.C., 1969, p. 509–20.
[95] Masehian E, Amin-Naseri M. A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning. J. Robot. Syst. 2004;21: 275–300.
[96] Khatib O. Real-time obstacle avoidance for manipulators and mobile robots. Int J Robot Res 1986;5:90–8.
[97] Sethian JA. Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science. Cambridge University Press; 1999.
[98] Kavraki LE, Svestka P, Latombe J-C, Overmars MH. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom. 1996;12:566–80.
[99] LaValle SM. Rapidly-exploring random trees: A new tool for path planning, TR 98-11, Computer Science Department, Iowa State University, 1998.
[100] Kavraki LE. Geometric methods in structural computational biology, CONNEXIONS, Rice University, available online from http://cnx.org/content/col10344/1.6/.
[101] De Berg M, Van Kreveld M, Overmars M, Schwarzkopf OC. Computational geometry. Springer; 2000.
[102] Bohlin R, Kavraki LE. Path planning using lazy PRM. In: Proceedings of IEEE international conference on robotics and automation, ICRA'00, 2000, p. 521–8.
[103] Amato NM, Bayazit OB, Dale LK, Jones C, Vallejo D. Choosing good distance metrics and local planners for probabilistic roadmap methods, In: Proceedings of IEEE international conference on robotics and automation, 1998, p. 630–7.
[104] Bessiere P, Ahuactzin J-M, Talbi E-G, Mazer E. The "Ariadne's clew" algorithm: global planning with local methods. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS), 1993, p. 1373–80.
[105] Zhang L, Manocha D. An efficient retraction-based RRT planner, In: IEEE International conference on robotics and automation, ICRA 2008, p. 3743–50.
[106] Khanmohammadi S, Mahdizadeh A. Density avoided sampling: An intelligent sampling technique for rapidly-exploring random trees. In: Eighth international conference on hybrid intelligent systems. His'08, IEEE; 2008. p. 672–7.
[107] Jackins CL, Tanimoto SL. Octrees and their use in representing three-dimensional objects. Comput Graph Inform Process 1980;14.
[108] Gonzalez-Badillo G, Medellin-Castillo HI, Lim T. Development of a haptic virtual reality system for assembly planning and evaluation. Procedia Technol 2013;7:265–72.
[109] Hsu D, Latombe J-C, Motwani R. Path planning in expansive configuration spaces. In: Proceedings of IEEE international conference on robotics and automation (ICRA), 1997, p. 2719–26.
[110] Plaku E, Kavraki LE. Distributed sampling-based roadmap of trees for large-scale motion planning. In: Proceedings of IEEE international conference on robotics and automation (ICRA), 2005, p. 3868–73.
[111] Wilson RH. A framework for geometric reasoning about tools in assembly, In: Proceedings of IEEE international conference on robotics and automation (ICRA), 1996, p. 1837–44.