Multibody Syst Dyn (2013) 29:189–211
DOI 10.1007/s11044-011-9294-3

Topological and dimensional synthesis of planar linkages for multiple kinematic tasks

Martín A. Pucheta · Alberto Cardona

Received: 17 April 2011 / Accepted: 5 December 2011 / Published online: 31 December 2011

Abstract This paper presents the combined use of two systematic methods for the synthesis of planar linkage mechanisms satisfying multiple kinematic tasks. First, a Graph Theory-based method is used to exhaustively enumerate the topological alternatives for a given problem. Then each feasible alternative is automatically dimensioned using the Precision Position Method; this computation includes space and design constraints. The existing methods to synthesize multiple tasks solve, in sequence, a decomposition of the problem into single kinematic tasks. The task decomposition and the topology selection for each task are usually performed by hand. This process leads to topologies with a repeated pattern and could lead to ignoring potentially desirable topologies. This paper analyzes a design strategy for the simultaneous solution of multiple kinematic tasks. This strategy has two advantages: (i) it eliminates the need for task decomposition, and (ii) it allows the exhaustive exploration of all non-isomorphic topologies up to a defined number of links. An example of simultaneous synthesis for a double rigid-body guidance task with application to a flap-tab mechanism is shown to illustrate the methodology.

Keywords Planar linkage mechanisms · Graph theory · Number synthesis · Modular dimensional synthesis · Multiple kinematic tasks · Synthesis strategies

1 Introduction

The synthesis of mechanisms consists in finding a suitable mechanism for given structural and functional requirements, including the design constraints (e.g., allowable space and minimum length of dimensions) and the motion of points or bodies to guide, defined as single or multiple kinematic tasks [10]. Single kinematic tasks can be classified into three main

M.A. Pucheta · A. Cardona (✉)
Centro Internacional de Métodos Computacionales en Ingeniería (CIMEC), INTEC - (Universidad Nacional del Litoral-CONICET), Güemes 3450, S3000GLN Santa Fe, Argentina
e-mail: acardona@intec.unl.edu.ar

M.A. Pucheta
e-mail: mpucheta@intec.unl.edu.ar

groups [10, 13]: function generation (FG), path following (PF), and rigid-body guidance (RBG). The prescription of multiple kinematic tasks and the allowable space constraint are two design requirements that frequently appear in industrial problems; however, techniques to solve them in a combined way have received little attention in the literature.

The type[1] and dimensional syntheses can be solved using either simultaneous or exhaustive methods. Three heuristic approaches can be mentioned among the *simultaneous* methods. In 1994, Fang [11] pioneered the use of Genetic Algorithms [18] to solve synthesis problems, including both the choice of the topology and its sizing. In 2007, Liu and McPhee [15] gave a genetic representation to the topologies and swept the topological space randomly; however, this approach ensures the complete exploration of the feasible topologies only if the number of function evaluations is high enough. Recently, Oliva and Goodman [20] achieved the selection of the best type for a given task by means of evolutionary algorithms and convertible agents; because they used a topological space defined by a four-bar and all six-bar, single degree-of-freedom (DOF) topologies, the randomness in the exploration of the topological space was eliminated. These techniques were applied for single tasks and without space constraints.

The *exhaustive* method, in contrast to the heuristic one, avoids missing any potentially useful alternative, as the topological alternatives are analyzed one-by-one in order. Most of the exhaustive methodologies first solve the type and number syntheses using Graph Theory concepts [35]. Then, for each topological option, several methods can be employed for the dimensional synthesis; for instance, Vucina and Freudenstein [36] used nonlinear programming, Raghavan [28] used iterative kinematic analyses, Sandor and Erdman [30] and Sardain [31] exploited Precision Position Methods, and more recently, Luo and Dai [16] used the Patterned Bootstrap method, which combines Precision Position and Homotopy methods. In this context, Pucheta and Cardona have proposed an algorithmic implementation called *Exhaustive Search Synthesis* [26], which consists of (i) an exhaustive enumeration of topological alternatives up to certain complexity and satisfying topological constraints [24], followed by (ii) a sizing of each feasible alternative using Precision Position Methods subject to space and other design constraints [23].

The typical strategy to tackle the multiple kinematic task problem is *to solve each single task in sequence*. Another possible strategy, explored and developed in this paper, is to generate and size the topologies that satisfy all the requirements at once; this process is called a *simultaneous synthesis*.

In this paper, the exhaustive search synthesis is applied to solve multiple kinematic tasks, and it is illustrated by applying it to the design of a flap-tab mechanism passing through three positions of a complex motion; see Fig. 1(a). For the sake of conciseness, the study is limited to the design of planar linkage mechanisms with simple joints. It is worth mentioning that the method can also be used to solve synthesis problems with multiple inputs.

A brief classification of the multiple kinematic tasks and a short description of the flap-tab problem are given in Sect. 2. The number and dimensional synthesis methods are reviewed in Sects. 3 and 4, respectively. Two synthesis strategies for multiple tasks are defined and illustrated in Sect. 5, and the results for a simultaneous synthesis case are presented and discussed in Sect. 6. The main conclusions are drawn in Sect. 7.

[1] Hereafter, the term "type" embraces both the type and the number synthesis stages of design.

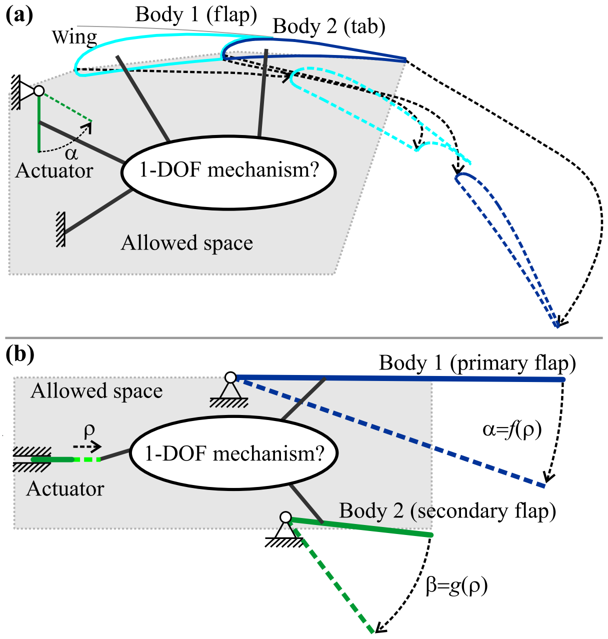


Fig. 1 Examples of multiple kinematic tasks: (**a**) Multiple rigid-body guidance; (**b**) Multiple function generation

2 Multiple kinematic tasks

A mechanism able to perform a single kinematic task (FG, PF, or RBG) has one input link (or joint) and one output link (or joint); thus, the task, and also the mechanism, can be classified as single-input-single-output (SISO). The classic four-bar linkage can generally develop one single task; however, if a single task has to be developed inside a restricted space, the solutions of the synthesis usually lead to six-bar [32] and eight-bar topologies. It is also worth mentioning that in pursuit of the simplest design of SISO mechanisms, the searches were often confined to the basic four-bar topology and the five mechanisms derived from the Watt and Stephenson kinematic chains [16, 20, 31, 32]. Within such a reduced topological space, all the possible ways of selecting the input and the output without repetitions could be performed by hand, but in the case of topologies with a higher number of links, this selection would be more complicated. A family of SISO n-bar linkages was recently studied by Chen and Angeles [5] for single motion generation tasks. In Sect. 3, the matching of the input and output parts for topologies with up to eight links will be done automatically.

Depending on the number of inputs and outputs, multiple kinematic tasks can be further classified into: Single-Input-Multiple-Output (SIMO, see Fig. 1), Multiple-Input-Single-Output (MISO), and Multiple-Input-Multiple-Output (MIMO) tasks.

The solutions for planar mechanisms developing multiple tasks are composed of multi-loop linkages. In previous works, Pucheta and Cardona solved SIMO kinematic tasks using a topological space confined to six-bar topologies, and applied them to: (i) the guidance of two points of a flexible trailing edge of an airplane wing, and (ii) the coordination of multiple flaps (Fig. 1(b)) in a turbine engine nozzle of divergent type [7, 23] and of convergent-divergent type [25]. Several design examples of SIMO linkages for multiple function generation can be found in [10]; nevertheless, the process used to select the topology is not described. Recently, Murray et al. [19, 21] proposed a methodology to design SIMO and MIMO n-bar linkages with application to shape changing mechanisms; these mechanisms

Table 1 Three precision positions of flap (N_1, E_1) and tab (N_2, E_2)

P.P.	N_1	E_1	N_2	E_2
	Disp. [m]	Rot. [°]	Disp. [m]	Rot. [°]
0	$\begin{bmatrix}0.0000\\0.0000\end{bmatrix}$	0.00	$\begin{bmatrix}0.0000\\0.0000\end{bmatrix}$	0.00
1	$\begin{bmatrix}0.6527\\-0.0173\end{bmatrix}$	−15.00	$\begin{bmatrix}0.7643\\-0.2071\end{bmatrix}$	−23.53
2	$\begin{bmatrix}0.7986\\-0.0230\end{bmatrix}$	−35.00	$\begin{bmatrix}0.8645\\-0.4439\end{bmatrix}$	−54.96

are composed by the successive addition of a repeated structure (e.g., a dyad) to a 4-bar linkage and were designed to satisfy three or more rigid-body precision positions.

2.1 A flap-tab mechanism

High-lift devices located on the trailing edge of an airplane wing are often made up of two parts, called the flap and the tab; see Fig. 1(a). The shape of the wing with the flap and tab when retracted is designed for cruise velocity. The flap and tab are deployed and rotated backward to increase the lift and the drag through increased camber and wing area at specific times, particularly during maneuvers of takeoff (lift is maximized) and landing (both lift and drag are maximized).

Due to their capacity to carry large loads, the rigid flap-tab mechanisms are currently preferred over those devices based on adaptable structures [4]. The existing patents of rigid flap-tab mechanisms can be divided into two broad categories: (i) fully-hinged linkages and (ii) slotted linkages, i.e., linkages with some straight or curved slider. Fully-hinged linkages are easy to actuate, and their low power consumption is an advantage; however, rigidity is difficult to ensure, and maintenance is crucial because of the many hinges. Slotted linkages are topologically simpler than the fully-hinged linkages and easier to design and control, but power consumption in the actuation systems (screw ball systems) and friction on the tracks are more significant. This research is restricted to the first category.

A rigid flap-tab mechanism develops a double rigid-body guidance task. In the mid-1970s, computer-aided linkage programs were used to solve this task; see, for instance, [30, pp. 288–289]. Nevertheless, the systematic exploration of the topological and dimensional design space for this problem is still a challenge.

2.2 Problem specification

For both the flap and the tab, trajectories and orientations at given configurations are prescribed by aeronautical requirements; in a first approximation, motion can be considered planar. The desired motions are discretized into three positions; these positions functionally correspond with *Folded* (cruise), *Take-off* and *Landing* positions; see Fig. 2. Node displacements and element rotations are shown in Table 1. The motorization on the existing revolute joint element must be computed; however, the three positions of the actuator could also be imposed (prescribed timing).

In the reference position, the absolute coordinates of the guided nodes of the flap and tab are $N_1 = (20.0481; 2.5553)$ and $N_2 = (20.5950; 2.6066)$, respectively; coordinates for the actuation point are (19.9000; 2.4700).

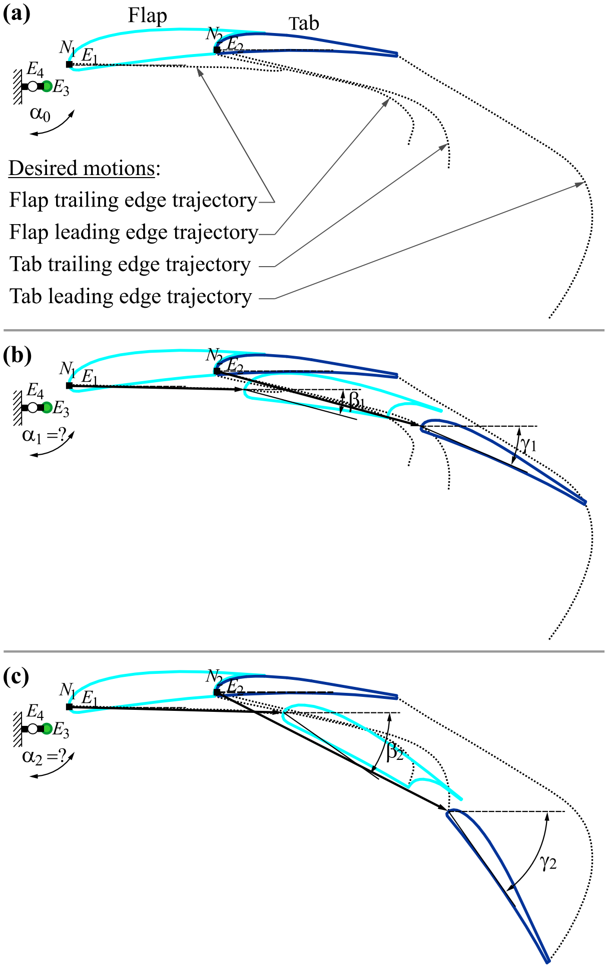


Fig. 2 Desired motions are shown in *black-dotted lines*; the three precision positions are: (**a**) reference position, (**b**) intermediate position, and (**c**) final position; the displacements for these positions are shown as *black arrows*; each line parallel to a reference configuration is shown in *dashed*

The complexity of this problem is increased by the restriction of space in which the mechanism must be moved. This *allowed space* constraint is a polygonal area defined by nodes with the following coordinates: (19.8000; 2.4700), (20.0481; 2.5553), (20.6410; 2.6221), (21.2657; 2.5879), (21.0000; 1.8000), (19.8000; 1.8000).

Note in the motions of Fig. 2 that the varying separation of the bodies during movement (imposed by aeronautical constraints) do not allow us to propose topologies where the flap and the tab are directly connected by a revolute joint. Otherwise, the method proposed by Murray et al. [19, 21] would be a good starting point for simplifying the topological solutions. Moreover, to provide stability, a curved slider between the flap and tab can always be added by computing the trajectory of a point on the tab relative to the flap.

3 Number synthesis

A graph representation of mechanisms, which is coherent with the one proposed by Tsai [35], is combined with matrix methods and algorithms for codification and isomor-

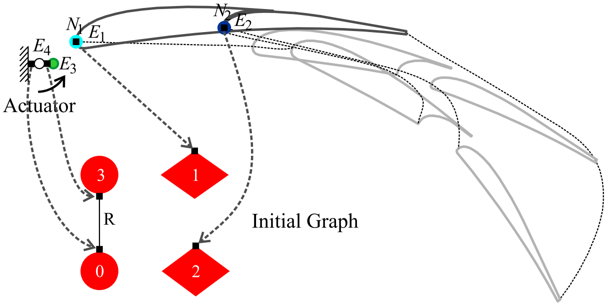


Fig. 3 Topological requirements for the flap-tab mechanism converted into a graph.

phism identification of mechanisms [23, 24]. The number synthesis stage consists of two manual steps (**T1** and **T2**) followed by three automatic steps (**T3**, **T4**, and **T5**):

T1: Represent the submechanism or initially prescribed parts and the kinematic task using a CAD preprocessor for multibody systems, discretizing the task into three or four precision positions.

T2: Select a desired atlas of mechanisms, denoted as A, with the desired number of degrees of freedom (desired number of inputs) and with specified types of links and joints. Define the topological constraints of the initial parts and desired mechanisms (explained below).

T3: Convert the multibody representation of the prescribed parts in a graph representation called *Initial Graph* G_{ini}, which describes mathematically the parts and structural restrictions.

T4: Solve the number synthesis by searching and identifying all nonisomorphic subgraph occurrences, inside each mechanism graph G_{A} in the selected atlas that coincide with G_{ini}.

T5: Create a schematic diagram for each mechanism.

3.1 Definitions

Some definitions are needed to explain the number synthesis algorithm and its parameters.

3.1.1 Graph of the kinematic problem for multiple tasks

The initial graph $G_{\text{ini}}(V_{\text{ini}}, E_{\text{ini}})$ consists of a set of vertices V_{ini} and a set of edges E_{ini}. Basically, bodies are represented by vertices of the graph and joints between bodies are given by edges connecting the vertices. Note that each isolated body to be guided has a single vertex representation in the graph of Fig. 3. An isolated joint would have been represented by an isolated edge, which is not admissible in Graph Theory. Therefore, in order to represent isolated joints, an auxiliary one-node body is introduced to connect the joint to the other part of the mechanism. For instance, the element E_3 in Fig. 3 is a one-node body used to impose a grounded revolute joint E_4 as the actuator.

3.1.2 Adjacency matrix

The adjacency matrix of the initial parts, denoted as A_{ini}, represents and stores the connectivity of the bodies, where entry ij is equal to one if body v_i is connected with body v_j and it is equal to zero otherwise. The adjacency matrix may represent any arbitrary kinematic chain composed of closed loops, open loops, and/or disconnected bodies. Because the initial parts can have disconnected bodies, A_{ini} can have both null rows and null columns. For example, the initial graph of the prescribed parts shown in Fig. 3 has the adjacency matrix

$$A_{\text{ini}} = \begin{array}{cc} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \end{array}, \tag{1}$$

indexed by the set of vertices $V = [v_0, v_1, v_2, v_3]$ labeled as $\mathcal{V}(V) = [0, 1, 2, 3]$ with the identifiers or names of the parts given in the problem description.

3.1.3 Type adjacency matrix

To represent mechanisms or submechanisms, a *type adjacency matrix* $\boldsymbol{T}$ is defined to assign link and joint types over the adjacency matrix: link types are assigned to the diagonal entries $\{0 = \text{ground}, 1 = \text{rigid}, 2 = \text{flexible}\}$, and joint types to the outer diagonal entries $\{1 = \text{revolute}, 2 = \text{prismatic}, 3 = \text{flexible\_hinge}, 4 = \text{clamped(fixed)}\}$. An atlas of mechanisms is built by the assignment of link and joint types to different kinematic chains in all nonisomorphic ways; each mechanism of the atlas is represented by a type adjacency matrix. Several atlases of simple-jointed linkage mechanisms can be obtained following the procedure explained in [24]. For example, the atlas *RigidOneDofR* is composed of single-DOF mechanisms with four, six, or eight rigid links, and joints of the revolute type. The type adjacency matrices of the graph G_{ini} and of the graph G_A of an eight-bar mechanism, which is used as an example below are, respectively,

$$\boldsymbol{T}_{\text{ini}} = \begin{array}{cc} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \end{array}, \quad \text{and} \tag{2}$$

$$\boldsymbol{T}_A = \begin{array}{cc} & \begin{array}{cccccccc} 0 & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \end{array} \\ \begin{array}{c} 0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{array} & \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}. \tag{3}$$

Note that the graph taken from the atlas in (3) has only one labeled vertex, the ground with label 0; the other vertices will be labeled in the number synthesis process.

Let us call $\boldsymbol{T}_A^p$ a reordering of $\boldsymbol{T}_A$ by a permutation vector p. Then a detected subgraph occurrence $G_{\text{ini}} \subset G_{\text{A}}^p$ can have the matrix representation

$$\boldsymbol{T}_A^p = \boldsymbol{T}(G_{\text{ini}}, G_{\text{A}}^p) = \begin{array}{c|cccccccc} & \mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} & 5 & 6 & 7 & 8 \\ \hline \mathbf{0} & 0 & 0 & 0 & \mathbf{1} & 1 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ \mathbf{2} & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ \mathbf{3} & \mathbf{1} & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 5 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 6 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 7 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 8 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{array}, \tag{4}$$

where vertices (highlighted in bold) identified with those of G_{ini} inherit its labels while new vertices (5, 6, 7, and 8) were labeled using new element IDs (ID number 4 is already in use by the joint element E_4).

A code-based identifier for symmetric matrices with integer entries is used for detecting isomorphic mechanisms. A vector code $C(\boldsymbol{T})$ is built for a given matrix $\boldsymbol{T}$ by concatenating row by row the diagonal and upper-right entries. Each row i, starting from the diagonal entry and ending at the last column, is codified into an integer number $C_i(\boldsymbol{T})$ using a numerical system in base b, where b is the maximum number of distinct entries among link types and joint types. (For instance, a base 2 or binary system is used for linkages with only revolute joints whereas a base 3 system is needed if prismatic joints are allowed, and so on). The code $C(\boldsymbol{T})$ is not unique. In order to obtain a unique identifier, all permutations of $\boldsymbol{T}$ are computed, and the maximum value of the vector (obtained by using a lexicographic comparison) is retained as the identifier. The number of required permutations are reduced by using structural properties of the graph associated to $\boldsymbol{T}$ [35]. For instance, permutations in groups of vertices with equal degree are performed, giving the identifier known as Degree Code and denoted as $DC(\boldsymbol{T})$ (more improvements on its computation, efficiency, and compact storage is given in [23, Chap. 2]). Two mechanisms, $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$, are isomorphic if $DC(\boldsymbol{T}_1) = DC(\boldsymbol{T}_2)$; this means that these matrices are equal for a particular permutation of one of them, and thus not only are their associated graphs isomorphic ($G_1 \cong G_2$), but their link and joint types are also equal. Each mechanism of the atlas can be stored using this identifier.

3.1.4 Synthesis adjacency matrix

A second matrix called the *synthesis adjacency matrix* $\boldsymbol{S}$ is defined to identify different subgraph occurrences by giving different meanings to diagonal entries belonging to each prescribed part (the off-diagonal entries are identical to those in the $\boldsymbol{T}$ definition). For instance, integer numbers 0, 2, 3, and 4 are assigned to ground, flap, tab, and actuator, respectively, while number 1 is assigned to each synthesized body:

$$\boldsymbol{S}(G_{\text{ini}}, G_{\text{A}}^p) = \begin{array}{c|cccccccc} & 0 & 1 & 2 & 3 & 5 & 6 & 7 & 8 \\ \hline 0 & \mathbf{0} & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & \mathbf{2} & 0 & 0 & 1 & 0 & 1 & 1 \\ 2 & 0 & 0 & \mathbf{3} & 1 & 0 & 0 & 1 & 1 \\ 3 & 1 & 0 & 1 & \mathbf{4} & 0 & 1 & 0 & 0 \\ 5 & 1 & 1 & 0 & 0 & \mathbf{1} & 1 & 0 & 0 \\ 6 & 0 & 0 & 0 & 1 & 1 & \mathbf{1} & 0 & 0 \\ 7 & 0 & 1 & 1 & 0 & 0 & 0 & \mathbf{1} & 0 \\ 8 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & \mathbf{1} \end{array}. \tag{5}$$

Matrix $\boldsymbol{S}$ gives a representation of the subgraph occurrence of G_{ini} inside a graph taken from the atlas G_{A}. Using the same code-based identifier used above but applied on $\boldsymbol{S}$, the operation $DC(\boldsymbol{S}(G_{\text{ini}}, G_{\text{A}}^{p1})) \neq DC(\boldsymbol{S}(G_{\text{ini}}, G_{\text{A}}^{p2}))$ identifies functionally different mechanisms, where G_{ini} is isomorphic to a subgraph of G_{A}^{p1} and of G_{A}^{p2}.

3.1.5 Additional data and topological constraints

Vertices marked with a diamond shape in Fig. 3 represent guided bodies, i.e., bodies with motions (displacements and/or rotations) defining a PF or RBG task. The significant dimensions of a planar linkage can be defined in several ways, for example, by means of a network of vectors connecting all joints. An additional vector is considered for each link performing a PF or RBG task; this vector connects any joint of that link with the guided point. Many of these vectors can be added to support multiple tasks of PF and RBG types.

The IDs of nodes with imposed trajectories are stored in a vector called $\boldsymbol{N}_{\text{traj}}$, and the IDs of their corresponding labeled vertices are saved in a vector called $\boldsymbol{V}_{\text{obj}}$; this information is used at the dimensional synthesis stage. The number of kinematic joints of a link defines its degree of connectivity. Also, the degree of each labeled vertex inside the initial graph is saved and used to reduce the number of unfeasible mechanisms in the subgraph search. A vector *minimum degree of vertices* deg_{min} of the initial parts is automatically identified from G_{ini}. In a similar fashion, a vector *maximum degree of vertices*, deg_{max}, could be imposed by the designer to define the connectivities of these bodies.

In the considered example, this additional data structure results:

$$\begin{aligned} \boldsymbol{N}_{\text{traj}} &= [1, 2]; \\ \boldsymbol{V}_{\text{obj}} &= [1, 2]; \\ \text{deg}_{\text{min}} &= \{\text{deg}(0), \text{deg}(1), \text{deg}(2), \text{deg}(3)\} = \{1, 0, 0, 1\}. \end{aligned} \tag{6}$$

3.2 Number synthesis by means of a subgraph search

The problem can be formally stated as the search of colored subgraphs isormorphic to G_{ini}, each of them denoted as H_A, included in graphs taken from an atlas A, that is,

$$\text{Find} \quad H_A \subseteq G_A, G_A \in A \quad / \quad H_A \cong G_{\text{ini}} \tag{7}$$

subject to the following constraints:

1. An equality of types constraint

$$\boldsymbol{T}(G_{\text{ini}}) = \boldsymbol{T}(H_A); \tag{8}$$

   i.e., the link and joint types in G_{ini} should match exactly those of the corresponding subgraph H_A of G_A;
2. A constraint on the degree of each vertex of the initial parts

$$\text{deg}_{\text{min}}[i] \leq \text{deg}(i) \leq \text{deg}_{\text{max}}[i] \quad \forall i \in \mathcal{V}(V_{\text{ini}}); \tag{9}$$

3. A constraint on the distance from each guided body to the ground

$$d_{\text{min}} \leq \min(\text{dist}(0, i)) \leq d_{\text{max}} \quad \forall i \in \boldsymbol{V}_{\text{obj}}, \tag{10}$$

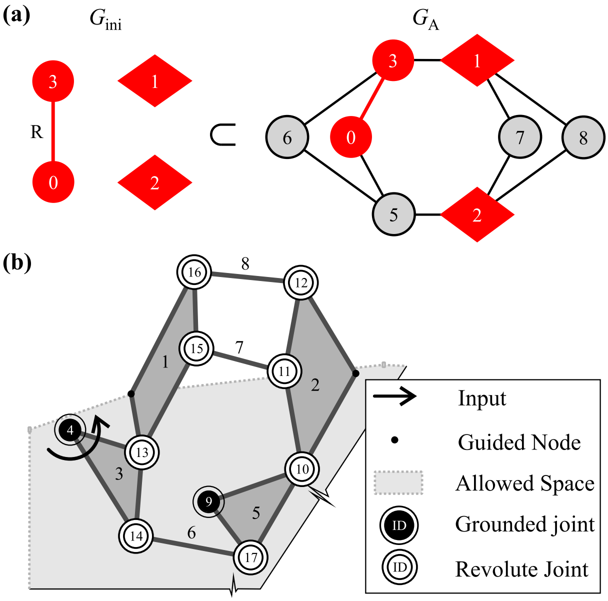


Fig. 4 A subgraph occurrence (**a**) with its schematic diagram (**b**)

where function dist(•, •) returns the length of the shortest path between both vertices; this constraint imposes the condition that distance from each guided body to the ground must be greater than or equal to $d_{\min}$ (a default value of 2 is used; a guided body connected to ground through a joint has distance 1, and this value has sense only when the guided body has to perform a single rotation or a circular path, which is a trivial case) and less than or equal to a limit $d_{\max}$ (a default value of $n_{pp} - 1$, with n_{pp} the number of precision positions, is used);

4. An isomorphism constraint, which imposes that the occurrence of G_{ini} in G_{A} must be functionally different from all previous answers; and
5. The pseudo-isomorphism constraint, which imposes that the solution G_{A} should not have a submechanism equal to a previously computed solution.

The subgraph search algorithm consists of a loop over the graphs G_A taken from the atlas A. Then, for each graph G_A an inner loop for taking all possible subgraphs H_A with $n_{\text{ini}} = |V_{\text{ini}}|$ vertices is developed, and the isomorphism test of (7) is executed. Several subgraphs satisfying that test can verify the equality of types constraint, (8), for a given graph G_A. Then the two sets of integer constraints, (9) and (10), reduce the number of unfeasible colored subgraph occurrences detected. A feasible mechanism solution M_K satisfying (7) to (10) is retained if it is functionally different from any previously stored mechanism M_I, $M_K \neq M_I$; $I = 0, \ldots, K - 1$. Finally, if it is required that the mechanism solution M_K must not contain a previous solution, the algorithm for computing the pseudo-isomorphism constraint executes one subgraph search for each previous solution M_I inside the current solution M_K, i.e., $M_I \not\subseteq M_K$; $I = 0, \ldots, K - 1$.

The default parameters for $\deg_{\min}$, $\deg_{\max}$, $d_{\min}$, and $d_{\max}$ can be defined by the user in terms of the IDs of nodes and elements of the existing initial parts.

Example 1: The atlas *RigidOneDofR* is selected as topological design space for the flap-tab problem passing through the three positions ($n_{pp} = 3$) stated in Fig. 2. A binary degree of the ground is required by imposing: $\deg_{\min}[0] = 2 \wedge \deg_{\max}[0] = 2$; the default parameters

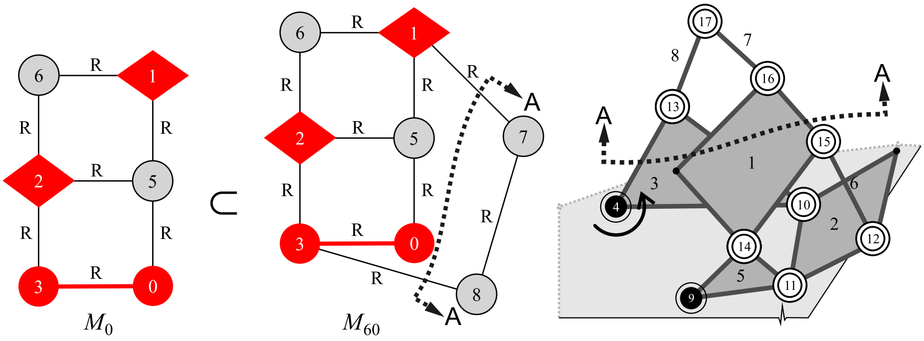


Fig. 5 Graph and physical meaning of a pseudo-isomorphic subgraph occurrence

of the distance constraint are used: $d_{\min} = 2 \wedge d_{\max} = 2$. A feasible solution is shown in Fig. 4(a); the mechanism has 8 links, 10 joints, and its type adjacency matrix corresponds to Eq. (4). Coordinates of known nodes are kept unchanged for plotting the sketch of Fig. 4(b).

Example 2: An example of occurrence of a pseudo-isomorphism is illustrated in Fig. 5 where $M_0 \subset M_{60}$. Links and joints out of the A-A cut in M_{60} are redundant because they do not add any new aspect to the behavior of mechanism M_0. Therefore, M_{60} is rejected by application of the pseudo-isomorphism constraint. Nevertheless, the redundant parallel subchains could be desired when planar stability, controllability, and a good distribution of forces and reactions are required; in this case, the constraint is not computed, and the computational cost of the algorithm is reduced.

4 Modular dimensional synthesis

The *Dimensional Synthesis* stage is performed with the combined use of a topology decomposition algorithm and a modular approach for the precision position method [23]. The following steps are computationally developed:

D1: Decompose the topology into single open chains (SOCs) that cover a minimal set of significant dimensions of the linkage.

D2: Search an order of the single-open chains solvable by the available SOC solvers (dyads and triads). Identify the variables of the problem: free parameters, coordinates of new pivots and multiplicity of each SOC. Default values for the bounds of the variables are automatically computed.

D3: Solve the chains analytically using a complex-number representation of links [13, 30]. If there are free parameters, a Genetic Algorithm is used to sweep the design space.

D4: Evaluate the fulfillment of the restrictions.

Between stages **D2** and **D3**, the user can intervene to modify bounds of variables and default settings of constraints parameters and of the Genetic Algorithm; otherwise, the execution is fully automatic [26].

4.1 Decomposition algorithm

In step **D1**, a decomposition algorithm is executed for splitting the topology into single open chains (SOCs). In general, this decomposition is not unique. The algorithm for searching

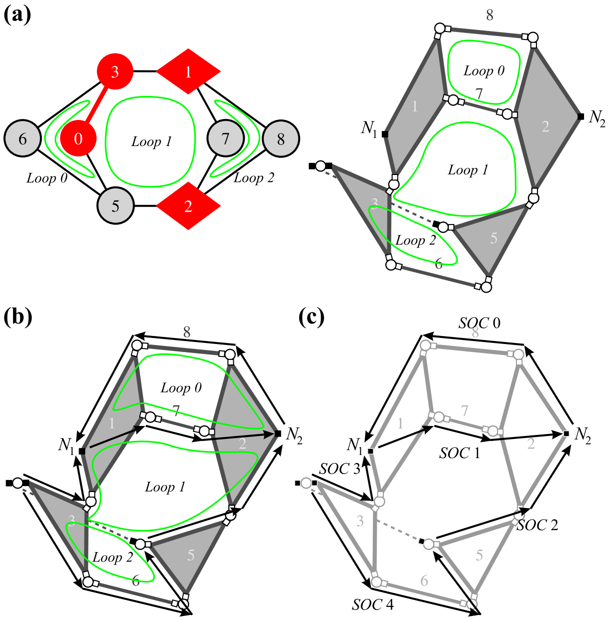


Fig. 6 (**a**) Basis of minimal independent loops of the graph and sketch shown in Fig. 4; (**b**) extension of the loops (by using $\boldsymbol{V}_{\text{obj}}$ and $\boldsymbol{N}_{\text{traj}}$) to pass through guided points N_1 and N_2; (**c**) a decomposition into five single open chains

the best set of SOCs uses a loop-per-loop decomposition that explores (i) the different bases of minimal independent loops (in some cases, the basis is not unique), (ii) the different combinations of loops for each basis, and (iii) the different loop orientations for each loop with a node with imposed trajectory.

The computation of the *basis of minimum independent loops* is made using the graph; see Figs. 4(a) and 6(a). These independent loops are used to find the significant dimensions of the mechanism. For PF or RBG tasks, where a guided body is present, the additional significant dimensions are found by means of the extension of the loop that visits a given vertex in $\boldsymbol{V}_{\text{obj}}[i]$ passing through the corresponding node in the $\boldsymbol{N}_{\text{traj}}[i]$ with equal index i; see Fig. 6(b). Each loop containing a guided body is divided into at least two SOCs; the two possible orderings for generating these SOCs, which can derive into different solvability conditions, is obtained by exploring both orientations of the corresponding loop in level (iii) of the search.

Given a loop-basis, a loop-order, and a loop-orientation, a Boolean classification of nodes is used to break the loops into SOCs; see Fig. 6(a). Nodes depicted with white squares □ are the *unknown* (or synthesized) nodes, while nodes depicted with black squares ■ are the nodes with *known* positions and imposed displacements. Positions of new pivots are unknown and must be computed at the dimensional synthesis stage; however, all nodes for the ground are considered as data for decomposition purposes: because they take part in pivots, their displacements are zero for every configuration. The loops are divided as follows: the first SOC starts in a known node (with known position and imposed displacements) and ends in another known node, between them there are only unknown nodes. Then all its underlying nodes are marked as known data using one auxiliary Boolean variable for each node, so that any of the following SOCs can start from any of these nodes. For example, loops of Fig. 6(b) are divided into SOCs of dyad and triad types as shown in Fig. 6(c). Loop 0 is divided into SOC 0 and SOC 1; Loop 1 is divided into SOC 3 and SOC 2; finally, Loop 2 results divided into SOC 4.

In step **D2**, for every SOC decomposition, a second part of the decomposition algorithm determines the solvability for each SOC by using the prescribed motions as initial data and simulating the transmission of motions from one SOC to the following one in sequence. The decomposition is qualified as feasible to be sized if all SOCs can be solved. For each feasible decomposition, we count the number of motion constraints that are solved; the decomposition that solves the most motion constraints in the sequence of SOCs is considered to be the best decomposition.

Steps **D1** and **D2** are repeated until all possible decompositions are finished; the best decomposition is retained for the dimensional synthesis stage. The variables are identified for the selected decomposition, and default values for the bounds of the variables are automatically computed [26]. If no decomposition is feasible to be solved for the available SOC modules, the topology is rejected.

Because the dimensional synthesis is based only on the Precision-Position Method, the automatic steps **D1** and **D2** are considered to be additional parts of the type synthesis solver, and the decomposition is executed for each feasible subgraph occurrence. For instance, the six-bar mechanism M_0 shown in Fig. 5 is decomposed using the available SOC solvers. However, for all decompositions, the result of the SOCs is over-constrained, and the topology is discarded. The topology with three loops and a binary ground shown in Fig. 4 is, among several subgraph occurrences, the first topology decomposable in the available SOC solvers and feasible to be sized for the prescribed motions. This topology is decomposed into five SOCs; see Fig. 6(c). Because three positions are imposed, each dyad (SOCs 2 and 3) will have a solution multiplicity of 2, while the triads (SOCs 0, 1, and 4) have a single solution. The coordinates of a new pivot and several missing motion constraints are automatically identified as variables of the problem.

4.2 Sizing algorithm

In steps **D3** and **D4**, an initial sizing solver can be executed for each feasible alternative.

The loop-closure equations for a mechanism passing through n_{pp} prescribed positions and decomposed into s ordered single-open chains that contain all its significant dimensions (as shown Fig. 6(c)), can be written as an ordered sequence of complex systems of equations in the form:

$$\mathbb{C}^k\mathbb{Z}^k = \mathbb{D}^k \quad k = 0, \dots, s-1 \quad \text{(no summation over } k \text{ implied)} \tag{11}$$

where the supra-index k denotes the SOC number or *computing order*. The complex matrix $\mathbb{C}$ contains entries that are functions of the link rotations; the complex vector $\mathbb{Z}$ contains the unknown links represented by complex numbers, and the independent term $\mathbb{D}$ contains entries which are functions of the displacements of the endpoints of the open chains and of the vector joining the endpoint of the open chain at the initial position. The loop-closure equations are solved by means of the execution of dyad or triad modules, recently presented in [26], and allow us to compute vectors $\mathbb{Z}^k$, $k = 0, \dots, s-1$ for a given number of prescribed positions and imposed motions. Because the loop-closure equations can present multiplicity of solutions and this multiplicity can be tabulated a priori, the solution procedure is hierarchically executed, thus all multiple solutions are combined.

If there are not free parameters, a loop over the several combinations of SOC solutions solves the system of (11). Otherwise, in the presence of free parameters, for each sequence

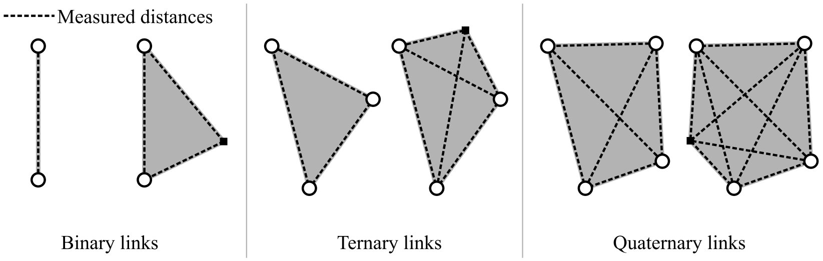


Fig. 7 Sizes of the links (with and without guided point) considered for the objective function

of SOC solutions a Genetic Algorithm (GA)[2] is used to find those free parameters that minimize an objective function subject to restrictions. The design space is defined by *variables* of different nature concatenated in vector $\boldsymbol{x}$. These include: coordinates of new pivots, missing motion constraints, and the set of free parameters, if there are any. Default values for the *bounds of variables* $\boldsymbol{x}_{\min}$ and $\boldsymbol{x}_{\max}$ are automatically computed. However, they can be modified by the user before starting sizing. That is, they can be modified immediately before starting step **D3**.

The *fitness* function $F(\boldsymbol{x})$ consists of a measure of the size of the mechanism $F^*(\boldsymbol{x})$ together with a contribution from constraints $Q(\boldsymbol{x})$ to the fitness function:

$$F(\boldsymbol{x}) = F^*(\boldsymbol{x}) + Q(\boldsymbol{x}), \tag{12}$$

The size of the mechanism $F^*(\boldsymbol{x})$ is computed as the summation of the sizes of the links; see Fig. 7.

The constraints contribution $Q(\boldsymbol{x})$ is a weighted measure of three terms: *minimal length of link dimensions* q_L, *noninversion of transmission angles* q_T (Fig. 8), and *allowed space constraint* q_A (Fig. 9), with weight factors λ_i that are adjusted empirically:

$$Q(\boldsymbol{x}) = \lambda_L q_L + \lambda_T q_T + \lambda_A q_A. \tag{13}$$

For every set of variables (an "individual"), the SOC modules are executed in sequence to compute the dimensions of the mechanism through (11). Then the individual with the *best fitness* will be the one that minimizes the mechanism size together with a better verification of constraints.

Because the problem is stated as a minimization, a high penalization parameter, $\lambda_{\max} = 1000$, is assigned if no solution exists for a given individual $\boldsymbol{x}$. Additionally, a characteristic length C_H is computed as the diagonal of the bounding box of all nodes defined in the initial problem. Using these two parameters, a *minimal length parameter* is defined as $L_{\min} = C_H/10$, and the weight factors are defined as $\lambda_L = \lambda_{\max}/2C_H$, $\lambda_T = \lambda_{\max}/2$, and $\lambda_A = \lambda_{\max}/C_H$.

The constraints are satisfied if their values are zero; their computation can be summarized as follows:

[2] Genetic algorithms are suitable for problems where neither domain nor goal function and restrictions are known; or they are so complicated that gradient computation becomes difficult or impossible [18]. Note that in this sizing problem, the system of (11) may not have a solution and it is difficult to compute the gradient of the space constraints.

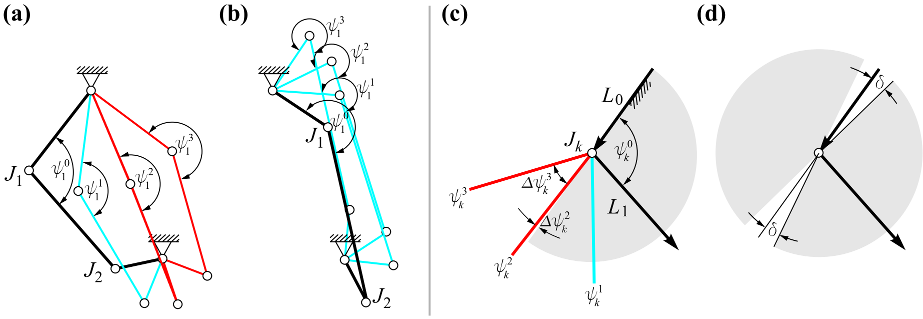
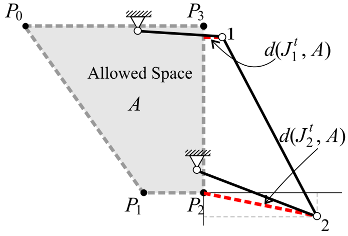


Fig. 8 Constraint of noninversion of transmission angles

Fig. 9 Violation of the allowed space constraint



- While the objective function is computed, the distances between nodes of the links are calculated. All of the distances must be higher than the minimal length parameter $L_{\min}$, and any violation of this restriction is accumulated in q_L.
- Figure 8 illustrates the effect of including the noninversion of transmission angles constraint. Situations like those shown in Fig. 8(a) are undesired and penalized, whereas those shown in Fig. 8(b) are desired. For any election of the input joint, the angle between the significant dimensions of the two links preserves its sign and does not surpass an angle of π during motion. For a given joint k, the angle of reference ψ_k^0 between the two links shown in Fig. 8(c) is used to define and select one of two allowed intervals of the motion from which the violations are computed. Refer to the grey sector in Fig. 8(c) to see these intervals. The motions falling outside the allowed interval, such as $\Delta\psi_k^2$ and $\Delta\psi_k^3$ in the example, are accumulated in q_T. To maintain distance from the aligned or singular configurations, the admissible intervals of relative rotations are reduced in an angle δ, as shown Fig. 8(d).
- The distances measured from each joint located outside the allowed space A are computed for every precision position. The highest value is accumulated as violation in $q_A = \max\langle d(J_i(t), A)\rangle$; $i = 1, \ldots, n_J$; $t = 0, \ldots, n_{pp} - 1$, where the function $d(\ ,\ )$ returns the outer-distance from the position of joint i at precision position t to the boundary of allowed space $A(P_0, P_1, \ldots, P_{n_A-1}, P_0)$, which is defined as a closed oriented polygon joining n_A points; n_J is the number of joints in the mechanism. The algorithm for computing the distance $d(\ ,\ )$ consists of a two steps procedure: (1) detect if the position of joint J_i is outside the allowed space, (2) if the joint is outside the allowed region, compute the penetration of that joint outside the region. The value of penetration is computed as the distance to the nearest segment of the polygon when the normal projection of the point falls within the considered segment; if no projection over a segment of the polygon exists, this value is computed as the distance to the nearest vertex of the polygon. For instance,

in the case shown in Fig. 9, the constraint takes a value equal to the distance from joint J_2 to the area A.

Eventually, instead of considering noninversion of transmission angles as a constraint, a full (more costly) kinematic analysis is conducted for each individual to compute the fitness function. For the chosen bounds of variables, the combinations of solutions between SOCs configure different problems, and thus different executions of the synthesis solver. For instance, the topology shown in Fig. 6(c) has two dyads with two solutions for each dyad, and three triads, each with a unique solution, resulting in 12 problems. From the 12 executions, the best qualified solution was saved. A solution can be qualified as *poor* or *bad* by considering (i) a marked violation of constraints, and/or (ii) branch and/or circuit defects.

We should remark that the capability of the presented method to perform with multiple tasks depends on:

1. The size (number of links and joints) of the mechanisms stored in the form of atlases. A larger number of inputs requires kinematic chains with a larger number of DOFs. A larger number of outputs requires larger kinematic chains, i.e., with more links and joints.
2. The available programmed SOC modules. Due to the use of the Precision-Position Method, if the kinematic chains are larger, the loops are longer and, therefore, the SOCs in which they are decomposed are longer. Therefore, higher order SOCs (e.g., open chains with four links, five links, and so on) are needed to solve larger kinematic chains.

5 Synthesis strategies

The optimal synthesis of a multiple-input-multiple-output mechanism involves two equally important steps: obtaining a good initial guess and optimizing the geometry for minimizing the kinematic errors. A good solution in the second step is strongly dependent on the quality of the initial guess. These steps can be repeated over subproblems by considering, for instance, one task per problem. Because tasks are solved in succession, we call this strategy *successive synthesis*. Alternatively, both steps can be developed once over the full topology. Because tasks are simultaneously computed in this second strategy, we call this strategy *simultaneous synthesis*.

To illustrate these concepts, let us consider the flap-tab example. Two possible synthesis strategies for designing such a mechanism are shown in Fig. 10.

1. The application of successive synthesis consists of one synthesis stage (number and dimensional) per task. After solving every task, an optimization stage can be executed to refine the required objectives and constraints. For example, first to find a mechanism which develops the flap motion, see Fig. 10(a-1); then, to complete the mechanism to develop the tab motion, see Fig. 10(a-2). For each mechanism, which is a solution of the first task, all nonisomorphic topological possibilities for the addition of subchains developing the second task must be considered. Another possibility is to design the tab first and then design the flap. If there are more than two bodies to guide, all orders must be considered.
2. The application of simultaneous synthesis is shown in Fig. 10(b). The required mechanism must connect all the prescribed parts and must satisfy the required motions. The computational cost of the dimensional synthesis stage is increased in comparison with the approach mentioned above. After the execution of number and dimensional synthesis, a unique optimization stage can be set up where all variables, objectives and constraints

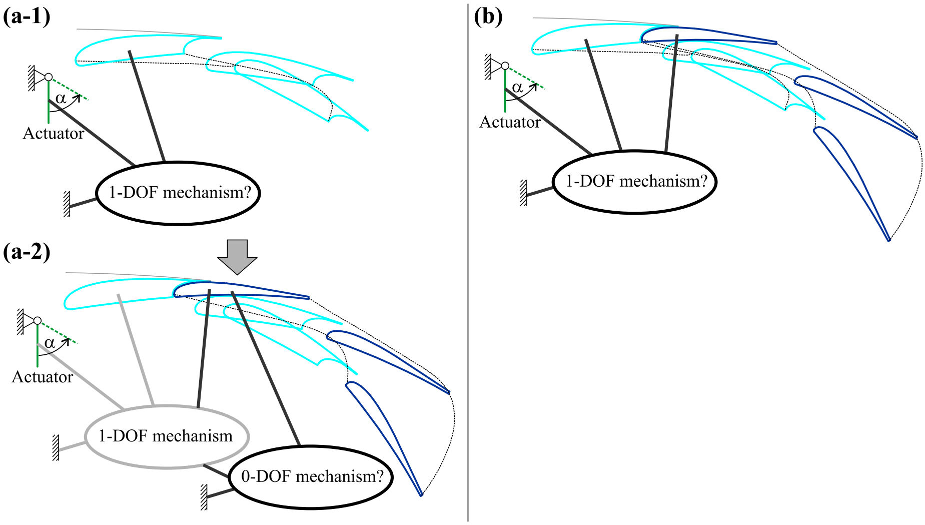


Fig. 10 Synthesis strategies: (**a**) successive; (**b**) simultaneous

are simultaneously considered. On the other hand, this approach avoids data transfer between subproblems and enables the designer to consider solutions that are neglected in the preceding approach.

From the topological point of view, solutions obtained from successive synthesis are limited to mechanisms formed by a single-DOF submechanism containing the ground and the input and used to guide one output body, followed by a zero-DOF submechanism including another output, which can also be connected to the ground through one or more pivots; see Fig. 10(a-2). The set of solutions obtained by simultaneous synthesis not only contain all topologies of successive synthesis, but also include other more complex topologies. Because a larger topological space is exhaustively swept, the possibilities of finding a suitable mechanism are increased.

Note that these two synthesis strategies are independent of the method employed for dimensional synthesis; analytical methods [30], homotopy-based methods [16, 32], and other geometric [17], heuristic [1, 3, 9, 22, 34], numerical, and optimization-based approaches [2, 5, 6, 8, 12, 14, 29, 33, 37] can be used.

The study presented in the next section is restricted to the search of a good initial guess by using the Precision-Position Method for each topological alternative.

6 Results

The successive synthesis strategy is the usual technique shown in textbooks and can be applied systematically and intuitively by an experienced designer. The focus of this research instead is to show that results of the simultaneous synthesis strategy include new nonintuitive topologies which cannot be obtained by the successive synthesis strategy, together with all those solutions given by the successive synthesis strategy. A simultaneous synthesis strategy for designing flap-tab mechanisms with a binary ground was employed as a first numerical experiment. User intervention is required between the execution of number and dimensional synthesis to define the bounds of the coordinates for the location of new pivots. All other steps are performed automatically without user intervention.

6.1 Number synthesis execution

An execution of the number synthesis solver was launched by requiring simple-jointed rigid mechanisms with joints of the revolute type (RigidOneDofR) up to topologies with 8 links and 10 joints, a binary ground, distance value of 2 (in terms of vertices of the graph) from ground to objective vertices, avoidance of pseudo-isomorphisms, and limiting the search to 100 subgraph occurrences.

The execution effectively resulted in 100 feasible subgraph occurrences, from which eighteen graphs were successfully decomposed into single-open chains. These feasible topologies start from kinematic chains with 8 links and 10 joints. The graphs and sketches of the first ten mechanisms are shown in Fig. 11. Note that the topologies of mechanisms 20 and 21 are subgraph occurrences of the initial graph found in the same mechanism of the atlas, but they are functionally different. In mechanism 20, the actuated body, link L_3, is connected to the flap, L_1, whereas in mechanism 21, link L_3 is connected to the tab, L_2. The same observation is valid for other topologies, e.g., 50 and 51, 52 and 53, 54 and 55, 56, and 57. Note also that the mechanism 20 does not have a 1-DOF submechanism containing the ground, the input L_3, and a guided body (either the flap or the tab). We remark that this linkage cannot be produced by a successive synthesis strategy. Moreover, all mechanisms obtained through the successive synthesis strategy are included in those obtained through the simultaneous synthesis enumeration; see, for example, mechanisms 50 and 51 in Fig. 11. Mechanism 50 has a Watt six-link 1-DOF kinematic chain guiding the tab, including links L_0, L_6, L_5, L_3, L_7, and L_2. The same type of chain is used to guide the flap in mechanism 51.

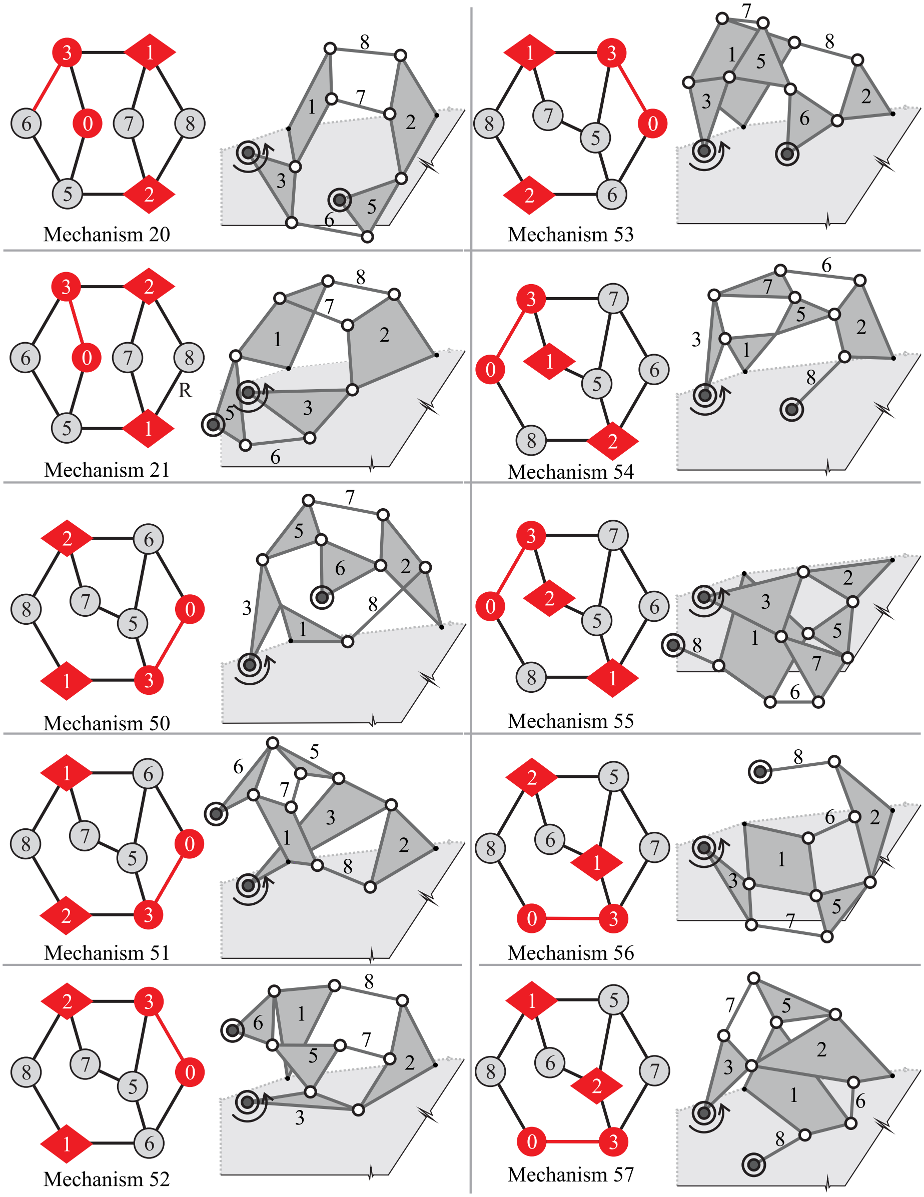
The topologies can be classified by the parameters used in the number synthesis, for instance, by the degree of the ground into binary, ternary, quaternary, and so on. From this classification and additional numerical experiments, which, for reasons of space they are not shown in this work, we observed that the more complex the ground is, the simpler the internal parts of the mechanism are. Thus, the relation between the number of pivots and the number of moving hinges gives an index of complexity of the moving parts.

6.2 Initial sizing execution

The parameters of the genetic algorithm used for the initial sizing solver were 90 individuals (10 individuals per variable), 120 generations, a probability of crossing of 0.5, and a probability of mutation of 0.01. The results were obtained in approximately 3 min per alternative, using a personal computer (CPU Intel Core 2 Duo 1.86 GHz with 2 GB RAM).

The best solutions found at the initial sizing stage were Mechanisms 20, 50, and 53, which are shown in Figs. 12 and 13. These solutions have the actuated body, L_3, connected with the flap, L_1.

- Alternative 20, in Fig. 12(a), showed the best fulfillment of the objective function and constraints. Link L_3 violates the area at the initial configuration.
- Alternative 50, in Fig. 12(b), has a small but unacceptable violation of the space by the hinge connecting links L_1 and L_8. Nevertheless, it could be corrected and avoided at a later optimization stage.



Mechanism 20

Mechanism 53

Mechanism 21

Mechanism 54

Mechanism 50

Mechanism 55

Mechanism 51

Mechanism 56

Mechanism 52

Mechanism 57

Fig. 11 Graphs and sketches for simultaneous synthesis with a binary ground constraint

- Alternative 53, in Fig. 13, showed a small violation of the minimal length constraint at links L_8 and L_3. In this figure, the continuum trajectories developed by the flap and tab between precision positions are also shown. One may clearly observe that the kinematic error of this solution needs to be further reduced; however, this mechanism is a good initial guess for using gradient-based optimization techniques [7].

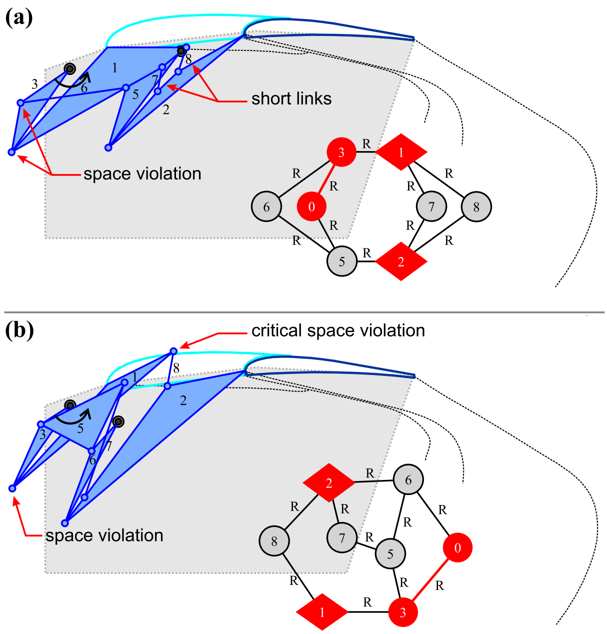


Fig. 12 Examples of mechanisms and constraints fulfillment: (**a**) Mechanism 20; (**b**) Mechanism 50

6.3 Future research

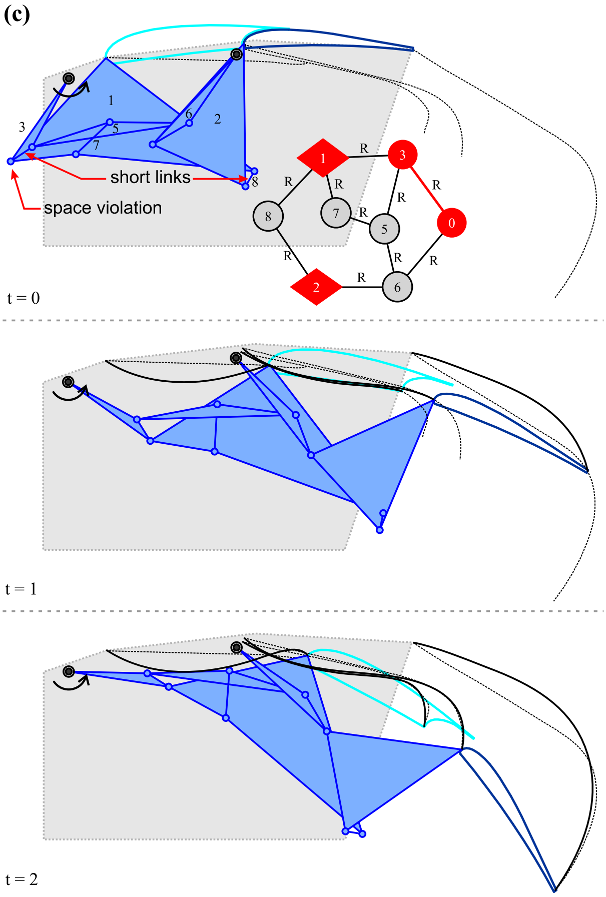
In future works, the kinematic error of the continuous tasks and the optimum transmission angles (with values near 90 degrees) will be included as objectives to further improve the quality of the solutions. This method can be combined with rigid-body replacement synthesis [27] to design partially compliant mechanisms performing multiple kinematic tasks with energy requirements. The method has the potential to solve the synthesis problem of spatial linkage mechanisms, by the inclusion of atlases of spatial topologies at the topological synthesis level and the inclusion of the appropriate solvers for spatial open chains, spatial transmission angles, and space constraints at the dimensional synthesis stage.

7 Conclusions

A method suitable for solving kinematic problems with multiple tasks was presented. Starting from the kinematic problem and a topological design space defined by a selected atlas of mechanisms, a two-step solver exhaustively enumerates nonisomorphic topologies that satisfy structural requirements, then automatically decomposes each topology into single-open chains, and finally sizes the mechanisms that satisfy the desired motion and space requirements. The topologies were obtained almost without human intervention. Each dimensioned topology constitutes, in most cases, a good initial condition for subsequent gradient-based optimization.

The proposed design strategy, called simultaneous synthesis of multiple kinematic tasks, has shown two advantages: (i) it eliminates the need for task decomposition avoiding human mistakes, and (ii) it allows the exhaustive exploration of all nonisomorphic topologies up to a given complexity related to the size of the atlas of mechanisms.

Fig. 13 Best qualified mechanism: Mechanism 53



The potential of the methodology for application to synthesis of real industrial problems has been shown throughout the paper by means of a double rigid-body guidance task. The eight-bar linkages obtained as solutions reflect the complexity of the proposed problem.

Acknowledgements This work has received financial support from the following Argentinean institutions: *Consejo Nacional de Investigaciones Científicas y Técnicas* PIP2009 112-200801-02473, *Agencia Nacional de Promoción Científica y Tecnológica* PICT-2010-1240, and *Universidad Nacional del Litoral* CAI+D 2009 PI65-330.

References

1. Acharyya, S., Mandal, M.: Performance of eas for four-bar linkage synthesis. Mech. Mach. Theory **44**(9), 1784–1794 (2009)
2. Avilés, R., Vallejo, J., Fernández de Bustos, I., Aguirrebeitia, J., Ajuria, G.: Optimum synthesis of planar linkages using a strain-energy error function under geometric constraints. Mech. Mach. Theory **45**(1), 65–79 (2010)
3. Cabrera, J., Ortiz, A., Nadal, F., Castillo, J.: An evolutionary algorithm for path synthesis of mechanisms. Mech. Mach. Theory **46**(2), 127–141 (2011)
4. Campanile, L.F.: Chapter 4: Lightweight shape-adaptable airfoils: A new challenge for an old dream. In: Wagg, D., Bond, I., Weaver, P., Friswell, M. (eds.) Adaptive Structures: Engineering Applications, pp. 89–135. Wiley, New York (2007)

5. Chen, C., Angeles, J.: A novel family of linkages for advanced motion synthesis. Mech. Mach. Theory **43**(7), 882–890 (2008)
6. Collard, J.F., Duysinx, P., Fisette, P.: Optimal synthesis of planar mechanisms via an extensible-link approach. Struct. Multidiscip. Optim. **42**, 403–415 (2010)
7. Cugnon, F., Cardona, A., Selvi, A., Paleczny, C., Pucheta, M.: Chapter: Synthesis and optimization of flexible mechanisms. In: Bottasso, C.L. (ed.) Multibody Dynamics, Computational Methods and Applications. Comp. Meth. in App. Sci., vol. 12, pp. 81–93. Springer, Amsterdam (2008)
8. Da Lio, M., Cossalter, V., Lot, R.: On the use of natural coordinates in optimal synthesis of mechanisms. Mech. Mach. Theory **35**(10), 1367–1389 (2000)
9. Diab, N., Smaili, A.: Optimum exact/approximate point synthesis of planar mechanisms. Mech. Mach. Theory **43**(12), 1610–1624 (2008)
10. Erdman, A.G., Sandor, G.: Mechanism Design: Analysis and Synthesis, 3rd edn., vol. 1, Prentice-Hall, New Jersey (1997)
11. Fang, W.E.: Simultaneous type and dimensional synthesis of mechanisms by genetic algorithms. In: Proc. of the 23rd Biennial Mechanisms Conference. Mechanism Synthesis and Analysis, vol. 70, pp. 35–41. ASME Design Engineering Division (1994)
12. Hansen, J.M.: Synthesis of mechanisms using time-varying dimensions. Multibody Syst. Dyn. **7**(1), 127–144 (2002)
13. Hartenberg, R.S., Denavit, J.: Kinematic Synthesis of Linkages. McGraw-Hill, New York (1964)
14. Jensen, O., Hansen, J.: Dimensional synthesis of spatial mechanisms and the problem of non-assembly. Multibody Syst. Dyn. **15**(2), 107–133 (2006)
15. Liu, Y., McPhee, J.: Automated kinematic synthesis of planar mechanisms with revolute joints. Mech. Based Des. Struct. Mach. **35**(4), 405–445 (2007)
16. Luo, Z., Dai, J.: Chapter: Searching for undiscovered planar straight-line linkages. In: Lenarčič, J., Roth, B. (eds.) Advances in Robot Kinematics, Mechanisms and Motion, Part 2, pp. 113–122. Springer, Berlin (2006)
17. McCarthy, J.M., Soh, G.S.: Geometric Design of Linkages, 2 edn. Springer, Berlin (2010)
18. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer, New Jersey (1997)
19. Murray, A., Schmiedeler, J., Korte, B.: Synthesis of planar, shape-changing rigid-body mechanisms. J. Mech. Des. **130**(3), 1–10 (2008)
20. Oliva, J.C., Goodman, E.D.: Simultaneous type and dimensional synthesis of planar 1DOF mechanisms using evolutionary search and convertible agents. ASME J. Mech. Robotics **2**(3), 1–9 (2010)
21. Persinger, J., Schmiedeler, J., Murray, A.: Synthesis of planar rigid-body mechanisms approximating shape changes defined by closed curves. J. Mech. Des. **131**(7), 1–7 (2009)
22. Peñuñuri, F., Peón-Escalante, R., Villanueva, C., Pech-Oy, D.: Synthesis of mechanisms for single and hybrid tasks using differential evolution. Mech. Mach. Theory **46**(10), 1335–1349 (2011)
23. Pucheta, M.: Computational methods for design and synthesis of planar mechanisms. Ph.D. thesis, Universidad Nacional del Litoral, Santa Fe, Argentina (2008)
24. Pucheta, M., Cardona, A.: An automated method for type synthesis of planar linkages based on a constrained subgraph isomorphism detection. Multibody Syst. Dyn. **18**(2), 233–258 (2007)
25. Pucheta, M., Cardona, A.: Synthesis of planar multi-loop linkages starting from existing parts or mechanisms: Enumeration and initial sizing. Mech. Based Des. Struct. Mach. **36**(4), 364–391 (2008)
26. Pucheta, M.A., Cardona, A.: Automated type and modular dimensional synthesis of planar linkages. In: Proc. of ASME IDETC/CIE Conferences. Montreal, Quebec, Canada (2010). Paper DETC-28540
27. Pucheta, M.A., Cardona, A.: Design of bistable compliant mechanisms using precision-position and rigid-body replacement methods. Mech. Mach. Theory **45**(2), 304–326 (2010)
28. Raghavan, M.: Number and dimensional synthesis of independent suspension mechanisms. Mech. Mach. Theory **31**(8), 1141–1153 (1996)
29. Sancibrian, R.: Improved GRG method for the optimal synthesis of linkages in function generation problems. Mech. Mach. Theory **46**(10), 1350–1375 (2011)
30. Sandor, G., Erdman, A.G.: Advanced Mechanism Design: Analysis and Synthesis, vol. 2. Prentice-Hall, New Jersey (1984)
31. Sardain, P.: Linkage synthesis: Topology selection fixed by dimensional constraints, study of an example. Mech. Mach. Theory **32**(1), 91–102 (1997)
32. Schreiber, H., Meer, K., Schmitt, B.J.: Dimensional synthesis of planar Stephenson mechanisms for motion generation using circlepoint search and homotopy methods. Mech. Mach. Theory **37**(7), 717–737 (2002)
33. Shen, Q., Al-Smadi, Y.M., Martin, P., Russell, K., Sodhi, R.: An extension of mechanism design optimization for motion generation. Mech. Mach. Theory **44**(9), 1759–1767 (2009)

34. Smaili, A., Diab, N.: Optimum synthesis of hybrid-task mechanisms using ant-gradient search method. Mech. Mach. Theory **42**(1), 115–130 (2007)
35. Tsai, L.: Mechanism Design: Enumeration of Kinematic Structures According to Function. CRC Press, Boca Raton (2001)
36. Vucina, D., Freudenstein, F.: An application of graph theory and nonlinear programming to the kinematic synthesis of mechanisms. Mech. Mach. Theory **26**, 553–563 (1991)
37. Wu, L.C., Carbone, G., Ceccarelli, M.: Designing an underactuated mechanism for a 1 active DOF finger operation. Mech. Mach. Theory **44**(2), 336–348 (2009)