Robotica (2024), 1–37
doi:10.1017/S0263574724001152

REVIEW ARTICLE

Optimizing soft robot design and tracking with and without evolutionary computation: an intensive survey

Fabio Stroppa[1], Fatimah Jabbar Majeed[2], Jana Batiya[1,†], Eray Baran[2] and Mine Sarac[1]

[1]Kadir Has University, Istanbul, Turkey
[2]Istanbul Bilgi University, Istanbul, Turkey
Corresponding author: Fabio Stroppa; Email: fabio.stroppa@khas.edu.tr

Received: 24 February 2024; **Revised:** 2 July 2024; **Accepted:** 2 August 2024

Keywords: design; modular robots; biomimetic robots; grasping; navigation; novel applications of robotics; redundant manipulators; serial manipulator design and kinematics; topological modeling of robots

Abstract
Soft robotic devices are designed for applications such as exploration, manipulation, search and rescue, medical surgery, rehabilitation, and assistance. Due to their complex kinematics, various and often hard-to-define degrees of freedom, and nonlinear properties of their material, designing and operating these devices can be quite challenging. Using tools such as optimization methods can improve the efficiency of these devices and help roboticists manufacture the robots they need. In this work, we present an extensive and systematic literature search on the optimization methods used for the mechanical design of soft robots, particularly focusing on literature exploiting evolutionary computation (EC). We completed the search in the IEEE, ACM, Springer, SAGE, Elsevier, MDPI, Scholar, and Scopus databases between 2009 and 2024 using the keywords “soft robot,” “design,” and “optimization.” We categorized our findings in terms of the type of soft robot (i.e., bio-inspired, cable-driven, continuum, fluid-driven, gripper, manipulator, modular), its application (exploration, manipulation, surgery), the optimization metrics (topology, force, locomotion, kinematics, sensors, and energy), and the optimization method (categorized as EC or non-EC methods). After providing a road map of our findings in the state of the art, we offer our observations concerning the implementation of the optimization methods and their advantages. We then conclude our paper with suggestions for future research.

1. Introduction

Robots and automated systems are transforming the way we live. They are designed and built in laboratories and research institutions, gradually transforming into industrial factories and eventually finding their way into our homes – liberating humans from monotonous, perilous, or unsafe responsibilities [1]. As such, they find extensive use in factory settings, where they are meticulously programmed to execute specific tasks with great efficiency [2, 3].

Conventionally, robotic systems are designed with rigid mechanical links and components. Such rigid and robust behavior allows them to interact with heavy and stiff objects without getting harmed or broken. Unfortunately, they possess many disadvantages. First, they are often heavy, cumbersome, and suffer from a lack of adaptability. Then, they exhibit limited performance in uncharted environments and when dealing with uncertainties [4]. Lastly, they might pose a risk to humans around them in the event of unintended or unexpected collisions – often leading to the segregation of robots from workplaces dominated by humans for safety.

The use of *soft robots* [5] might eliminate most of the disadvantages of rigid robots. They are composed of soft-compliant materials and joints that enable them to bend, contract, stretch, and deform.

[†]Formerly

Their structure resembling biological systems allows for safe interaction between humans and machines. They possess unique features such as twisting with high curvature, so they are ideal for functioning in tightly constrained environments. Finally, their compliant mechanisms allow for adaptability, which simplifies many tasks (e.g., grasping) and improves mobility [1]. They are used in a wide range of applications including reaching or squeezing through a small aperture [6], grasping [7], rehabilitation, and artificial hands [8], search and rescue [9], and haptic rendering [10]. While using soft robots has great potential, they have certain limitations that need to be resolved: (i) their motion results in many infinite degrees of freedom (DoFs) – significantly complicating their kinematics design and control strategies, and (ii) because their soft/compliant behavior can be obtained by different approaches, a set of general design guidelines and modeling methods is needed [11].

Ultimately, it is essential for roboticists to systematically model and design soft robots to exploit their full potential. Designing soft robots usually requires special attention, particularly on two main aspects: their control and their morphology. First, they must allow for easy control and actuation, which can be quite challenging due to the nonlinear properties of their soft material. Second, they can adapt to tasks by deforming and changing their shape to orient themselves in uncharted environments. During their operation, these deformations and shape changes are mostly estimated rather than accurately measured due to the lack of sensorization technologies and techniques. Thus, managing the soft robots' morphology while maintaining its control is challenging because it is hard to predict how a soft robot will deform and move.

The challenges of soft robots' design can be addressed using mathematical optimization – finding a particular set of decision variables that meet specific constraints while minimizing or maximizing one or more objectives regarding their design. Such optimization tools maximize the effectiveness of soft robots to meet the needs of a desired application and offer a solution to their complex kinematics problem. This way, we could exploit soft robots in the workplace to their full potential without hindering reachability constraints. Linear programming [12], integer programming [13], nonlinear programming [14], quadratic programming [15], stochastic programming [16], and dynamic programming [17] are all commonly used optimization techniques; however, in recent years, an increasing number of researchers have turned to evolutionary computation (EC) [18]. EC is a branch of artificial intelligence that specializes in optimization problems with algorithms inspired by natural evolution [18]. Compared to other traditional optimization methods, EC is more robust [19] and flexible [20] and can easily handle complex task spaces [21]. It is often used for optimizing rigid robotic designs such as exoskeletons [22], and consequently, it is a promising approach for soft robot design as well. EC encompasses a large variety of algorithms, each with specific strengths and weaknesses depending on the objective functions and constraints. As the boundaries among EC algorithms are blurry, choosing the right algorithm for soft robot design optimization might be a dreadful process.

In this work, we will provide a review of the state-of-the-art optimization-based design of soft robots while answering the question: "which optimization algorithm should an engineer use for the design of a soft robot?" Toward this goal, we will first present a new definition of what we considered a *soft robot* in our study, classify the types of soft robots along with their possible applications, and list the metrics that are optimized in the literature (see Section 2). Then, we will analyze different optimization algorithms while highlighting their corresponding advantages and disadvantages for each soft robot, particularly focusing on EC in Section 3. Next, we will compare and evaluate the performance of miscellaneous optimization algorithms, considering their impact on the design of soft robots (see Section 4). Lastly, we will provide the reader with a summary of past studies and recent findings that can serve as a guideline for future soft-roboticist designers (see Section 5).

1.1. Search strategy and eligibility criteria

In this literature survey, we specifically focused on how different optimization methods were used for designing the mechanical structure of soft robots by:

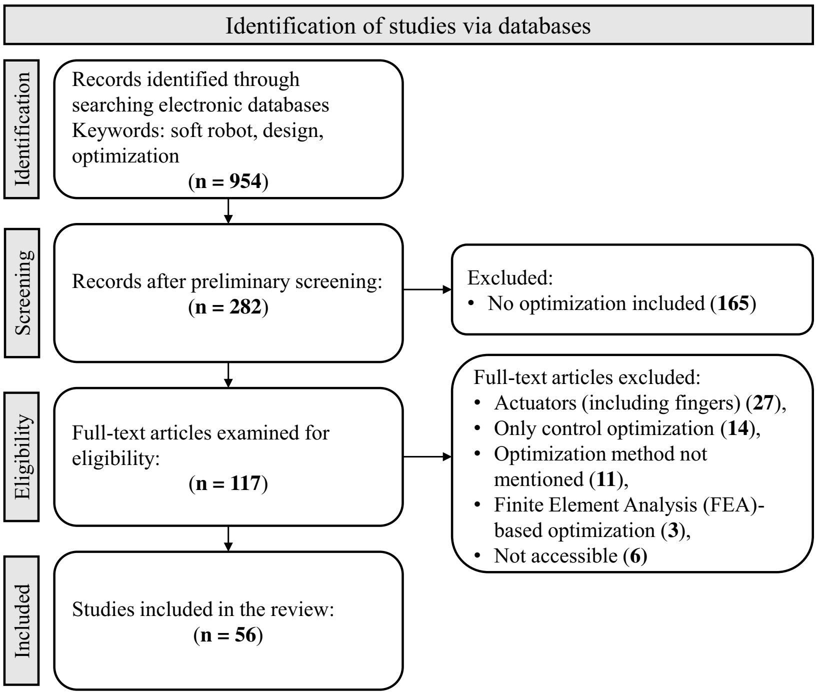
Identification of studies via databases
Identification
Screening
Eligibility
Included
Records identified through searching electronic databases Keywords: soft robot, design, optimization (n = 954)
Records after preliminary screening: (n = 282)
Excluded: • No optimization included (165)
Full-text articles examined for eligibility: (n = 117)
Full-text articles excluded: • Actuators (including fingers) (27), • Only control optimization (14), • Optimization method not mentioned (11), • Finite Element Analysis (FEA)-based optimization (3), • Not accessible (6)
Studies included in the review: (n = 56)

Figure 1. *Flow diagram of the systematic literature review protocol.*

- retrieving the best combination of parameters allowing the robot to achieve an optimal design for one of more specific metrics (e.g., the best lengths for its link in the kinematic chain, the best configuration that allows the robot to maximize the traveled distance during locomotion, or the best configuration of sensors to explore the environment); and
- solving the inverse kinematics problem rather than using classical methods (e.g., geometry or Denavit–Hartenberg) since the innovative design of the specific soft robot affected its kinematics.

Focusing only on mechanical design optimization allowed us to provide a more accurate and detailed discussion. Therefore, we eliminated the following topics from this survey – although also related to soft robot optimization:

- studies in which the mechanical design of a soft robot was developed without optimization, and optimization methods were only used after the design process and did not affect the design;
- studies on soft actuators only with no complete application scenario;
- studies presenting soft fingers with no complete application scenario – even if presented as a first procedural step of designing soft grippers in the future;
- studies focusing on control – unless the design was affected by it; and
- studies focusing on structural optimization via finite element analysis, since it evaluates how different materials respond to various force interactions and stress/strain scenarios without changing the topology/kinematics decisions.

Figure 1 summarizes the search and elimination criteria. We completed the literature search in the IEEE, ACM, Springer, SAGE, Elsevier, MDPI, Scholar, and Scopus databases from 2009 to 2024 (January) using the keywords "soft robot," "design," and "optimization." We investigated 954 studies

Table I. *List of acronyms for type of robot categories.*

Type of robot	Acronym	Type of robot	Acronym
Bio-inspired	Bio	Cable-driven	Cab
Continuum	Con	Fluid-driven	Flu
Gripper	Gri	Manipulator	Man
Material-property actuated	Mat	Modular	Mod

and filtered out the ones for which the main focus was not the mechanical design of soft robots through optimization methods, along with a hundred overlaps obtained from Scopus.

2. Background

2.1. Background on soft robots

Soft robots differ from conventional rigid, mechanical robots by being flexible and deformable. However, since flexibility and deformability can be achieved in many ways, this is too vague of a definition for soft robots. Instead, we propose the following definition: *"soft robots are mechanical systems that cannot be modeled through rigid links and joints and are made up of soft or flexible structures."* For example, a robot made up of extremely small rigid parts that are connected to form a serial structure such that a robot seems to have compliance is not regarded as soft because it is not composed of soft joints or links. In the following sections, we will categorize soft robots based on their (i) type, (ii) application, and (iii) design metric/parameter to optimize.

2.1.1. Types of soft robots

Table I summarizes the common types of soft robot types and their acronyms to be used in the rest of the text. Figure 2 shows some examples of soft robots found in the literature, including bio-inspired robots [23, 24], modular robots [25, 26], grippers [27, 28], and continuum robots [29, 30]. Note that these categories are not mutually exclusive: some soft robots can combine these categories or their technologies to achieve specific functionalities according to their task.

Bio-inspired soft robots mimic the structure, motion, functionality, or behavior of living mechanisms, such as fish [31], snakes [32], worms [33], quadrupeds [34], lizards [35], and insects [36]. They can exhibit a variety of locomotion styles found in nature, such as swimming, crawling, flying, or slithering – depending on the living creature they are mimicking. This allows the robots to navigate through complex environments with great agility and adaptability. Their drawback is the complicated control schemes they require to achieve specific movements precisely. Additionally, some methods might be energetically demanding to accomplish bio-inspired locomotion, which can be improved through optimization techniques

Cable/tendon-driven soft robots use tension cables as their main actuation mechanism [37, 38], pulling the robot body to bend, stretch, and deform. They are usually lightweight, compact, compliant, and capable of safe interaction. They can be particularly useful for creating fine and precise movements in applications such as medical operations – thanks to their distributed actuation. Their drawbacks include the need to replace and maintain the cables as they wear out with time and usage. Additionally, controlling the joint movements is more difficult compared to traditional rigid robots due to the distributed cable system throughout the actuated section of the robot body.

Fluid-driven soft robots use fluidic pressure, such as pneumatic or hydraulic pressure, for actuation [39, 40]. The actuators fill up with the fluid medium to change the shape of the soft robot and achieve motion and manipulation. Due to the fluids' inherent properties, they are highly compliant and can achieve safe and soft interaction with the environment. Moreover, they can adapt well to unstructured and dynamic environments. Their drawback is mostly the complexity of the control

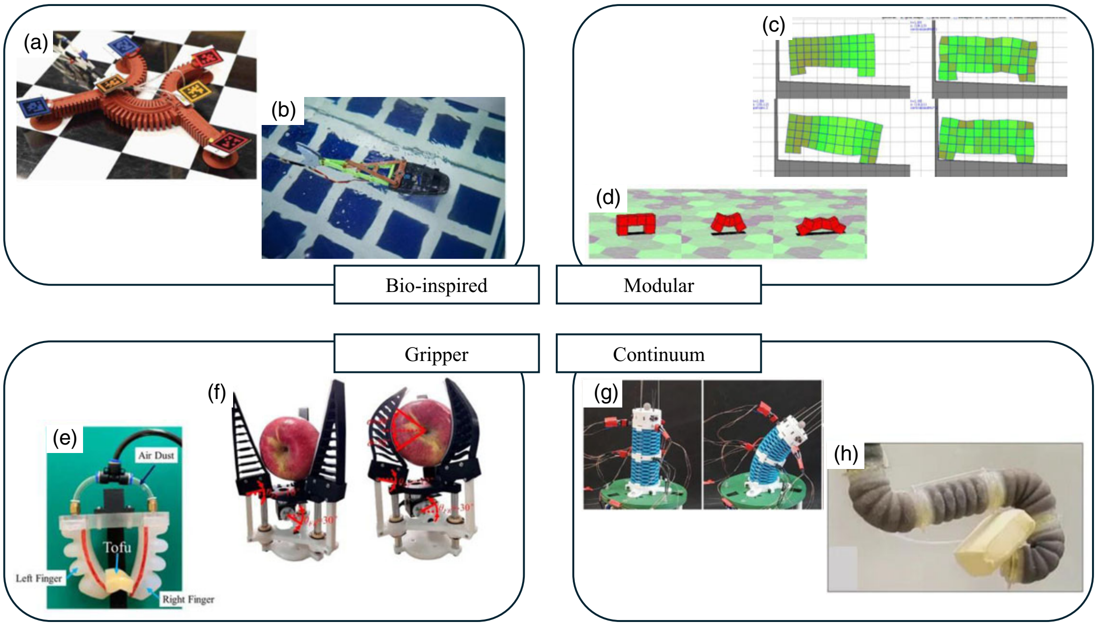

Figure 2. *Some examples of common soft robot types found in the literature: bio-inspired (a) gecko-like soft robot [23] and (b) Soft swimming fish [24]; modular soft robots (c) [25] and (d) [26]; soft grippers (e) [27] and (f) [28]; and continuum soft robots (g) [29] and (h) [30]. All images were used under CC-BY license.*

system due to the nonlinear behavior of fluid dynamics. Additionally, despite their capability of achieving quick movements, the speed of fluid-based soft robots may not always match that of rigid-link robots, especially in applications that require high-speed precision.

Soft grippers are an adaptable class of end effectors with flexible and soft materials that can grasp objects of different sizes and shapes. They usually consist of soft fingers [41], soft-granular material [20], or magnetic-based inflatable systems [42]. Their advantages include manipulation with a gentle, humanlike touch, robustness, and compliance. Depending on their actuation mechanism, the complexity of the control scheme can vary. One of their drawbacks is having a limited payload – achieving the necessary force for grasping different objects can be challenging. Additionally, complex designs might be required that can adapt to grasping a wide range of various objects, and additional sensors might be needed.

Soft manipulators, as opposed to conventional rigid manipulators, are designed with soft materials – such as fabric [42, 43], plastic [39], elastomers [40, 44], or flexible polymers [45] to produce compliant movements.

Continuum soft robots are manipulators that resemble the movements and structures of organisms found in nature, such as octopus tentacles [46] or elephant trunks [47]. Their continuous bodies can perform smooth and continuous movements, such as bending and moving seamlessly without discrete segments. They are typically made up of elastic and flexible segments connected. They offer distributed actuation, similar to fluid and cable-driven soft robots, and are capable of performing multi-DoF motions. Continuum robots can be considered as manipulators. The drawback of these devices is the complexity of the control and modeling problem imposed by their continuum nature.

Material-property-actuated soft robots are actuated by altering the properties of the materials they are composed of, which are responsive to an external stimulus. The materials are selected to respond to a specific type of external stimuli, thus actuating the robot in the desired fashion. There is a wide variety

Table II. *List of acronyms for applications of soft robots.*

Application of robot	**Acronym**
Exploration/locomotion/tracking	ELT
Grasping/manipulation	M
Medical surgery and procedures	MSP

of materials that can be used to manufacture this category of soft robots, including piezoelectric, magnetically responsive, heat sensitive, light sensitive, and chemical materials. The selection of a suitable material can be done according to the application of the robot and the way the robot is intended to move. For instance,

- Piezoelectric materials induce controlled mechanical deformations as a response to electrical currents [47] to provide precise and rapid reversible movements. They can either be embedded into the entire structure of the robot or fixed to specific regions. They are great at offering precise and rapid actuation in complex movements, thus quick and controlled movements. They are generally small in size and lightweight and can be used in soft robots without limiting flexibility. However, most of these materials are prone to mechanical fatigue and thus permanent deformation after being exposed to high strain magnitudes.
- Magnetically responsive materials are embedded or coated on soft materials, which respond to the magnetic field and produce various types of movements (i.e., bending, twisting, or extending) [33, 48]. The type of movement is determined by the direction, pattern, and intensity of the magnetic particles and field. Their soft nature and precise motion ability are extremely useful for delicate tasks in confined and complex environments (e.g., for surgical applications). Moreover, they do not require additional external power cables, rendering the overall system lightweight and small in size. However, they feature the drawback of a complex control algorithm with precise calibration and a limited speed. Additionally, the response of the magnetic particles may be heterogeneous due to the material structure, and maintaining a uniform response across the robot's structure may be challenging.

Modular soft robots are composed of individual modules that can be interconnected and reconfigured to create diverse robots with different functionalities [49, 50]. Each module is independent from the other modules and self-contained with an independent control unit, power cell, sensors, and actuation mechanism. When these independent modules are combined, the modular soft robots can adapt to different shapes and environments. They provide flexibility in design that creates novel structures to suit the task, scalability (i.e., the robot might be scaled up or down by adding or removing modules), increased robustness due to the increased redundancy (i.e., such that a malfunctioning module does not result in a complete breakdown of the robot), and ease of maintenance (i.e., a problematic module can be replaced individually without the need to disassemble the entire robot). Their drawbacks include an increased complexity in the control scheme to achieve seamless coordination between the modules and the need for standardization of communication and control protocols among the modules.

2.1.2. Soft robot applications

The applications of soft robots can extend to a variety of different fields. We classified them into three main categories as summarized in Table II with their acronyms that will be used in the rest of the text.

Exploration/locomotion/tracking applications take advantage of soft robots thanks to their adaptability and soft/compliant nature. They are widely used for the exploration of unknown terrains and

Table III. *List of acronyms for optimization metrics.*

Optimization metrics	**Acronym**
Topology	T
Kinematics	K
Force	F
Locomotion	L
Sensors	S
Energy	E

cluttered environments such as narrow spaces [43] or search and rescue operations in dangerous and/or debris-filled regions that traditional rigid robots may find challenging to navigate [51].

Grasping and object manipulation applications are well-suited for soft robots since they can deform to grasp and move objects or manipulate the environment safely due to their flexible and soft nature. These properties allow them to bend and accommodate the shape of the object to be grasped and thus moved easily. This can effectively be performed by soft robots on delicate and soft objects [42, 52], for multi-object grasping [53], or for picking up samples from hazardous environments [54].

Medical surgery and procedure applications can take advantage of soft robots due to their soft structures [55, 56]. In these applications, soft robots travel through and manipulate different parts of the human body without causing harm and injury to the internal organs, unlike rigid robots. It is worth mentioning that soft robots designed for medical surgery and procedure applications can be perceived as a subfield of grasping and manipulation robots with a special aim.

2.1.3. Optimization metrics

Table III summarizes the classification of the metrics and aspects of soft-robot designs that are optimized in the literature with acronyms that will be used in the rest of the text.

Topology of the design is directly altered to yield the optimal shape of the robot through a comprehensive exploration of the design possibilities in terms of structural configurations, material selection, and transducers to maximize the overall performance of a soft robot. This can include the geometry, configuration of the robot, morphology, and center of gravity. To facilitate this, designers can employ advanced computational design and simulation tools for analyzing and predicting the responses of different topological configurations. Optimizing the topology is key to expanding the capabilities of a soft robot and intelligently making use of its flexibility.

Kinematics of the design refers to the relationship between the angles of deformable joints, linkages, and geometric configurations, which contribute to understanding the range of motion (i.e., the workspace of the robot) and maximizing it. Obtaining the kinematics of a robot often requires complex modeling of the geometric structure and the segmentation of the robot, along with simulations, as well as optimization methods to understand and predict the complex interactions among the joints, actuators, and flexible structures. A well-defined kinematics relationship ensures whether a specific motion is achievable as well as finding the most efficient way to achieve it. However, when the robots feature soft components or innovative DoFs such as growth and retraction, the kinematics become more complex and different than the ones of rigid robots [57].

Output forces of the soft robot design can be maximized (e.g., grasping force in the case of grippers) through optimization methods. While optimizing the forces for the soft robots, achieving a certain output force is challenging because many factors need to be taken into account, such as the nonlinear nature of soft materials, the selection of actuation methods to obtain the desired force, the geometry of the soft structure, the control architecture, and many other aspects to attain and amplify the available forces.

Locomotion of the soft robot design can be optimized to achieve a correct balance among material properties, control schemes, and actuation mechanisms. To obtain efficient and versatile movements, it is crucial to overcome the challenges imposed by the soft materials and actuation methods – such as actuation type, speed, locomotion gaits, control type, and power management [58]. Designers must exploit the nature of the flexible materials to make robots adaptable to different terrains, such that future research will expand the applications in this area.

Sensors of the soft robot design can be optimized in terms of placement, type, and number. This optimization is crucial for maximizing the adaptability and performance of soft robots. This is due to obtaining more useful data from sensors to be used in purposes such as control feedback [19] as well as kinematics and shape estimation in real time while eliminating the need for complex models of soft robots [59].

Energy optimization in soft robot design is essential for enhancing the efficiency and sustainability of the systems to be designed. That is, the total energy consumption can be minimized during deformations of soft robots to obtain efficient and precise movements in various tasks. Moreover, considering the energy consumption during the design stage can provide soft robotic systems with an extended operational life.

2.2. Background on optimization and evolutionary computation

2.2.1. Optimization problems

Optimization is the process of solving a problem expressed as in Eq. (1). Specifically, it consists of finding a particular set of *decision variables* (x) that meet specific constraints (inequalities g_j, equalities h_k, and bounds) while minimizing or maximizing one or more *objective functions* (f_m).

$$\begin{aligned} \text{minimize/maximize} \quad & \{f_1(x), \ldots, f_M(x)\} \\ & x = \{x_1, \ldots, x_N\}^T \\ & x_i \in x, \quad i = 1, \ldots, N \\ \text{subject to} \quad & g_j(x) \geq 0, \quad j = 1, \ldots, J \\ & h_k(x) = 0, \quad k = 1, \ldots, K \\ & x_i{}^L \leq x_i \leq x_i{}^U \end{aligned} \tag{1}$$

When more than one objective function is involved in the optimization problem, we define it as a multi-objective optimization problem (MOOP). In most cases, all these objectives conflict with each other, meaning that the optimization of one objective happens to the detriment of another one. This makes it difficult to optimize one objective without worsening others. Therefore, instead of obtaining a single optimal solution, the MOOPs require the retrieval of *trade-off* solutions, which are referred to as the *Pareto* optimal front [60, 61]. These solutions that lie on the Pareto optimal front are *non-dominated* by other solutions – meaning no other solution is better in all objectives. On the other hand, trade-off solutions might be optimal for one objective but not for others [62, 63].

2.2.2. Optimization methods: exact versus approximate methods in engineering

Algorithms or procedures solving optimization problems, namely, *optimization methods*, can be mainly classified as *exact* or *approximate* methods. Exact methods retrieve *the* exact optimal solution to a problem [64], whereas approximate methods obtain suboptimal solutions that are approximations of the exact optimal solution. Engineers favor the latter to solve design problems due to the complexity of the objective functions. Designing robotic devices – such as soft robots – is a complex task that usually involves a considerable amount of variables and parameters. This makes the effectiveness of exact methods not always guaranteed [65], especially considering that these methods are gradient-based and require the objective function to be differentiable [66] – which is mostly not the case in real engineering problems. Furthermore, reliable and robust solutions are preferred over exact ones: slight changes in the value of the optimal solution may result in overall instability [67], and uncertainty in the search space may lead to unfeasible optimal solutions [68].

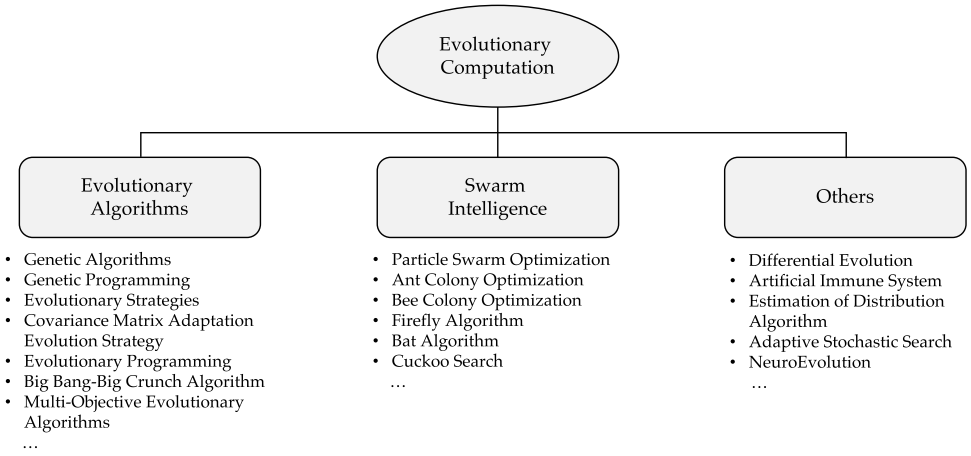

Figure 3. *Evolutionary computation taxonomy with some of the most relevant algorithms.*

While exact methods are usually based on math, most of the approximate methods are based on *heuristic* search [69]. A heuristic is an estimation of how close a point in the search space is to the optimum, and it can be expressed in many forms such as geometrical distance in the space or state evaluation functions. While many approximate methods are heuristic-based, under certain conditions, heuristic methods can guarantee optimality and therefore be considered exact – for example, the path planning algorithm A* [70] can find the optimal path if its heuristic is admissible and consistent [71].

2.2.3. Evolutionary computation

EC is a branch of artificial intelligence focusing on solving optimization problems based on some of the most popular approximate techniques in engineering [22, 62] – Figure 3 shows a partial taxonomy of EC methods. EC is mainly inspired by Darwin's theory of natural selection, in which individuals adapt to their habitat to ensure the survival of the species – therefore, EC techniques are considered as *meta*heuristic algorithms. EC techniques mimic this process by randomly generating a population of potential solutions (*individuals*) to an optimization problem and then gradually "evolving" them toward the optimal one. In algorithmic terms, *individual evolution* simply consists of mixing and changing numerical values to decision variables until better solutions (i.e., with higher/lower value of the objective function) are found. The specific process depends on the algorithm's specifics, and EC encompasses a wide range of algorithms [18, 72, 73]. The flowchart depicted in Figure 4 shows the generalized framework of any EC algorithm (applicable to evolutionary algorithms, swarm intelligence, etc.).

Because EC techniques propagate several solutions concurrently rather than just one, they fall under the *population-based* methods [74]. This property allows algorithms to obtain several optimal solutions when needed, which applies to multimodal (i.e., when also local optima are sought in addition to the global one) or conflicting MOOPs (i.e., when more than one objective is involved). This property also allows EC to parallelize the process and cover a larger area of the search space while avoiding premature convergence [75].

Lastly, engineers are also drawn to EC techniques because they simplify the formalization of problems. These techniques can be applied to non-differentiable functions, work without the need for precise mathematical equations in their objectives, and are effective regardless of the mathematical formulation of the problem. This makes the methods independent from the specific optimization problem, such that it is possible to utilize the same technique in diverse scenarios. This is particularly helpful in mechanical design, where the design process might be dependent on a model simulation [76].

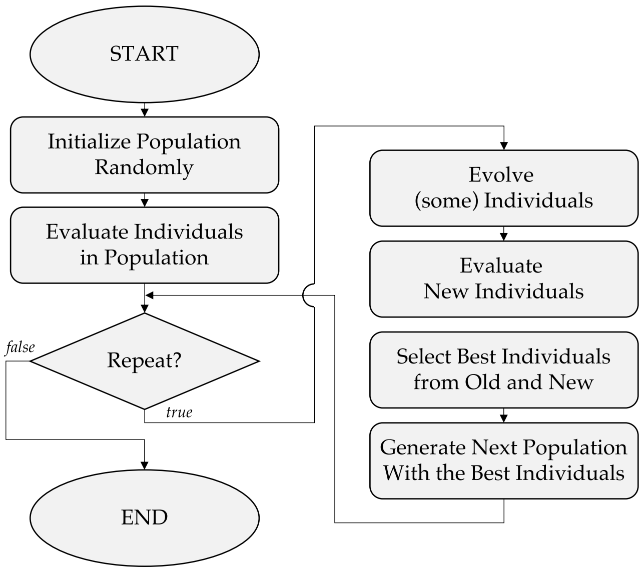
START
Initialize Population Randomly
Evaluate Individuals in Population
Repeat?
false
true
END
Evolve (some) Individuals
Evaluate New Individuals
Select Best Individuals from Old and New
Generate Next Population With the Best Individuals

Figure 4. *Generalized flowchart framework of any algorithm in evolutionary computation.*

3. Optimization of soft robots design in literature

We have reviewed the state-of-the-art in soft robot design optimization, with a primary focus on the diverse methodologies and approaches employed in the definition and solution of optimization problems. The advantages of EC prompted us to categorize our findings into two groups: studies using EC techniques (Section 3.1) and studies using other methods (Section 3.2). In both categories, our analysis delved into the specific types of soft robots and their intended applications, the optimized metrics (i.e., the objective function(s)), and the optimization method.

3.1. Soft robot designs with evolutionary computation techniques

Table IV reports the list of EC techniques used in the studies we evaluated, along with their abbreviation and a reference to the original article where the method was proposed, whereas Table 5 shows the studies where EC techniques were used to optimize output force, kinematics, locomotion, topology, sensors, and energy consumption. We observed that the most commonly used EC techniques in soft-robot design optimization are evolutionary/genetic algorithms, particle swarm optimization, matrix adaptation evolution strategies, and neuroevolution.

3.1.1. Evolutionary algorithms (EAs) and genetic algorithms (GAs)

EAs, particularly GAs [77, 78], rank among the most popular EC techniques. They mimic the natural process of selection and the survival of the fittest [78]. These algorithms (i) create a group of random solutions within the problem's search space, referred to as a *population* of *individuals* or *chromosomes* to resemble genetic processes, (ii) assign a *fitness* value to each solution by evaluating the objective function, and (iii) generate new solutions by combining the values of individuals from the current population, a process known as *crossover*. By allowing only the fittest individuals to undergo crossover and be retained in future *generations* (i.e., the following iterations of the algorithms), GAs direct their population toward the optimal solution (i.e., evolve). The crossover operation exploits the strengths of good solutions (i.e., the most fitting), facilitating fasting convergence to a (sub)optimal solution. However, this process does not guarantee the retrieval of *the* global optimum and may get stuck to a local one. To address this issue, GAs incorporate a genetic *mutation* operator, inspired by natural mutation, which

Table IV. *Evolutionary computation techniques.*

Abbreviation	Algorithm
GA	Single-Objective Genetic Algorithm [77, 78]
NSGA-II	Elitist Non-dominated Sorting Genetic Algorithm [79]
NSGA-III	Reference-Point Based Elitist Non-dominated Sorting Genetic Algorithm [80]
SPEA2	Strength Pareto Evolutionary Algorithm 2 [81]
PSO	Particle Swarm Optimization [82]
DE	Differential Evolution [83]
EDA	Estimation of Distribution Algorithm [84]
CMA-ES	Covariance Matrix Adaptation Evolution Strategy [85, 86]
NEAT	NeuroEvolution of Augmenting Topologies [87]
CPPN-NEAT	Compositional Pattern Producing Network with NEAT [88]
MO-NEAT	Multi-objective NEAT [89]
SE	Speciated Evolver [90]
NS	Novelty Search [91]
AFPO	Age-Fitness-Pareto Optimization [92]
ASS	Adaptive Stochastic Search [93]
MFF	Multi-objective Maximin Fitness Function [94]
RP	Multi-objective Rank Partitioning [57]

randomly alters the values of a newly generated solution. This promotes exploration of the search space and allows the algorithm to escape from local optima. Additionally, some GA implementations retain the most fitting individuals in the population, such that they can be used in the following generation for crossover to exploit promising regions of the search space (a strategy known as *elitisim*); other implementations do not allow any individual of the previous generation to be retained in the new one – that is, the new generation is only composed of the offspring generated by crossover and mutation – allowing for a more extensive exploration of the search space albeit storing the most fitting individual separately from the population (a *non-elitist* strategy).

Apart from the benefits of population-based techniques (see Section 2.2.3), GAs offer the advantages of being simple to implement and highly effective in converging toward (sub)optimal solutions. The biggest disadvantage is that their many genetic operators come with many parameters, and fine-tuning these parameters can be challenging – often done by trial and error. Furthermore, due to their iterative stochastic nature, GAs are not an efficient choice for real-time optimization.

3.1.1.1. Single-objective genetic algorithms. GAs were first proposed to solve single-objective optimization problems. Although there are different versions (e.g., binary or real coded, elitist or not, and many variants for the genetic operators), the main process is the same as the one described previously. There are even readily available software packages for nonexpert users of these methods: for example, MATLAB Optimization Toolbox offers an implementation of GA with the command ga.

In our literature review, we found the following research studies that used GAs to optimize the design of soft robots:

- Rieffel et al. [106] used a GA to find the optimal morphology (actuators, joints, and materials) of a soft pneumatic robot by maximizing the distance traveled per gait period;
- Hiller and Lipson [105] used a GA to optimize the topology of a voxel-based soft robot by maximizing the number of voxel moved during a locomotion task;
- Dinakaran et al. [107] used a GA to optimize the design of a soft gripper, specifically to tune the stiffness, thus the corresponding load capacity, depending on the actuation pressure;

Table V. *Designs with evolutionary computation techniques.*

Authors	Type of robot	Application	Metrics	Optimization Method
Sui et al. [95]	Mod, Flu	ELT	L	GA [77, 78]
Berger et al. [96]	Mat	ELT	T, L	GA [77, 78]
Marzougui et al. [97]	Mod	ELT	L	GA [77, 78]
Abbaszadeh et al. [31]	Bio	ELT	K	GA [78, 98], PSO [82]
Bodily et al. [99]	Con, Flu, Man	ELT, M	K	GA [77, 78] with MFF [94]
Stroppa [57]	Flu, Con, Man	ELT, M	K	GA [77, 78] with RP [57]
Fathurrohim et al. [100]	Bio	ELT	F	GPR [101] with GA [77, 78] + HC [102]
Liu et al. [103]	Gri, Flu	M	T, F	NSGA-II [79]
Fitzgerald et al. [7, 20]	Gri, Flu	M	F, T	NSGA-III [80]
Kriegman et al. [104]	Mod	ELT	L	GA [77, 78] with AFPO [92]
Hiller and Lipson [105]	Mat, Mod	ELT	T, L	GA [77, 78]
Rieffel et al. [106]	Con, Flu, Bio, Cab	ELT	L	GA [77, 78]
Dinakaran et al. [107]	Gri, Flu, Mat	M	F	GA [77, 78]
Chikhaoui et al. [108]	Cab, Con	MSP	T	PSO [82]
Djeffal et al. [109]	Cab, Con	ELT	K	PSO [82]
Merrad et al. [110]	Cab, Con	ELT	K	PSO [82]
Chen et al. [111]	Mod, Man	M	K	PSO [82]
Gao et al. [112]	Cab, Flu	M	K	PSO [82]
Atia et al. [47]	Bio, Mat, Mod	M	T	PSO [82]
Cheong et al. [113]	Con	M	K, F	EDA [84]
Tan et al. [114]	Cab, Con, Gri	M	K	DE [83, 115]
Ferigo et al. [19, 116]	Mod	ELT	S	CMA-ES [85, 86]
Medvet et al. [90]	Mod	ELT	T, L	CMA-ES [85, 86], GA [77, 78], NEAT [87] with SE [90]
Kimura et al. [50]	Mod, Flu	ELT	T, L	MO-NEAT [89] with SPEA2 [81] + NSGA-II [79]
Cheney et al. [6]	Mat, Mod	ELT	T, L	CPPN-NEAT [88], MO-NEAT [89]
Methenitis et al. [117]	Mat, Mod	ELT	T, L	CPPN-NEAT [88] with NS [91]
Exarchos et al. [118]	Flu, Con, Man	ELT, M	K	ASS [93]

- Sui et al. [95] used a GA to optimize the deformation process of a soft robot by tuning its locomotion velocity (it is unclear whether they are minimizing or maximizing their objective function);
- Berger et al. [96] used a GA to optimize the design of a mesh-based soft robot that moves through vibration by maximizing the traveled distance in a locomotion task;
- Marzougui et al. [97] used a GA to optimize the morphology of a voxel-based soft robot by maximizing the distance it traveled in a single dimension during a locomotion task;

- Abbaszadeh et al. [31] used a GA (and a PSO) to optimize the strain gauge of a soft-aquatic robot for kinematics function regression;
- Medvet et al. [90] used GA to optimize the morphology of a voxel-based soft robot with bio-mimicking motion by maximizing the distance it traveled during a locomotion task; and
- Fathurrohim et al. [100] used a GA to tune the hyperparameters of a neural network (Gaussian process regression [101], see Section 3.1.7), for optimizing the stiffness of a soft-robot fish's fin by maximizing the time-averaged thrust force it produces.

3.1.1.2. Multi-objective evolutionary algorithms. EAs are widely used also in MOOPs [75, 119] – namely, multi-objective evolutionary algorithms (MOEAs). This is mostly due to their population-based nature, which allows for multiple solutions to be retrieved at once – this is convenient in MOOPs, where we seek different trade-off solutions distributed on the Pareto optimal front. Although many different MOEAs exist in the state of the art, the most famous ones (in both usage and efficiency) are the Elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II [79] and the following version NSGA-III [80]) and the Strength Pareto Evolutionary Algorithm 2 (SPEA2 [81]). Both of them are efficient MOEAS in terms of convergence to the Pareto optimal front and diversity in the retrieved non-dominated solutions. NSGA implements a sorting procedure for solutions by iteratively forming ranks dominance and uses the crowding distance operator to guarantee diversity in the population, whereas SPEA2 ranks the solutions based on their strength (i.e., how many solutions are being dominated by the one in analysis) and guarantees diversity in the population with a truncation operator (i.e., selects non-dominated solutions to be removed from the population based on how close they are to other non-dominated solutions). There are even readily available software packages for nonexpert users of these methods: for example, MATLAB Optimization Toolbox implements a variant of NSGA-II with the command `gamultiobj`, whereas SPEA2 is usually self-coded by researchers.

In our literature review, we found the following research studies that used these two MOEAs to optimize the design of soft robots:

- Liu et al. [103] used NSGA-II to maximize the reaction force (thus, grasping performance) while designing a gripper with the minimum offset of the center of gravity.
- Fitzgerald et al. [7, 20] used NSGA-III in two studies to optimize the grain morphology of soft gripper by maximizing the pull-off force when gripping spherical objects of three different sizes.
- Kimura et al. [50] used SPEA2 for designing voxel-based soft robots through multi-objective NEAT (see Section 3.1.7); however, they modified it by adding the crowding distance of NSGA-II (i.e., the niching operator that guarantees diversity in the Pareto set).

Besides NSGA-II, NSGA-III, and SPEA2, an alternative way of implementing an MOEA is to scalarize the set of objectives into a single objective. The simplest and most common way is to multiply each objective by a user-supplied weight, namely, the weighted-sum method [61]. In our literature review, we found two studies implementing two different approaches:

- *Multi-objective Maximin Fitness Function* (MFF) [94], a strategy to formulate the fitness function such that it directs the GA generations that are both close to the Pareto optimal front and diverse and used by Bodily et al. [99] to optimize the design of a soft-pneumatic robot by maximizing its (i) dexterity and (ii) load-bearing capacity; and
- *Rank Partitioning* (RP), proposed for and implemented directly in soft-robot optimization by Stroppa [57], in which each objective is associated with a different priority, and sorted based on it, such that each individual of the population is grouped in ranks and sub ranks; this method was used to optimize the design of a soft growing robot (i.e., solve the inverse kinematics) for a

specific task by minimizing the error between the end effector and desired target, in tasks having multiple targets and obstacles.

3.1.1.3. Age-fitness-Pareto optimization (AFPO). The AFPO [92] is a method for avoiding premature convergence in single-objective EAs by exploiting multi-objective optimization strategies. In addition to the (single) main objective (i.e., the objective function or the fitness), each individual is also evaluated based on how long it has been in the population (i.e., its age). Specifically, the age objective is minimized – the state of the art shows that partitioning an evolving population into age groups can greatly improve the ability to identify global optima and avoid converging to local optima [120, 121].

In our literature review, we found the following research study that used AFPO to optimize the design of soft robots:

- Kriegman et al. [104] used a GA with AFPO to optimize the shape of a voxel-based soft robot for locomotion by maximizing the distance it traveled during a locomotion task.

3.1.2. Swarm intelligence (SI) and particle swarm optimization (PSO)

SI is an EC technique inspired by swarms, herds, or flocks of animals gathering to locate food or build colonies [122]. The population of individuals – in this case, the *swarm* – features a collective behavior and follows the trend of the most fitting individual, exploiting its information for convergence. The most used method in SI is PSO [82], which mimics a flock of birds changing their direction based on the one locating food, forming a choreography of individuals pointing at the optimal solution. Each individual in PSO changes its search pattern by learning from the rest of the swarm's behavior. This cooperative and iterative process makes the swarm converge to the optimal solution. When compared to EAs, PSO is computationally inexpensive in both time and space, and it often shows better performance, a higher convergence rate, and higher-quality solutions [123, 124]. PSO also features different multi-objective optimization versions [119], which however suffer from poor convergence to the Pareto front and with insufficient diversity.

In our literature review, we found the following research studies that used PSO to optimize the design of soft robots:

- Abbaszadeh et al. [31] used PSO to optimize the strain gauge of a soft-aquatic robot for kinematics function regression;
- Chikhaoui et al. [108] used PSO to optimize the curvature of two robotic arms by maximizing the number of collaborative configurations among the ones retrieved with forward kinematics;
- Djeffal et al. [109] and Merrad et al. [110] used PSO to optimize the accuracy of the kinematic computation for the continuum robot's design by minimizing the displacement between the end effector and the target;
- Atia et al. [47] used PSO to optimize the design of an elephant-trunk-inspired soft robot by minimizing the differences between the desired robot shape and the current one;
- Chen et al. [111] used PSO to optimize the path planning and trajectory tracking through inverse kinematics of a modular manipulator (could be either soft or rigid) by minimizing its maximum joint bending angles at any moment; and
- Gao et al. [112] used PSO to optimize the accuracy of the kinematic computation for a hybrid manipulator by minimizing the displacement between the end effector and the target.

3.1.3. Differential evolution (DE)

DE is used for problems having continuous domains. Individuals are represented as vectors, and perturbations are applied with vector arithmetics to diversify the population and explore the search space [83].

However, when dealing with MOOPs, studies have shown that DE is less efficient than other MOEAs as it does not converge to the Pareto optimal front but to a nearby suboptimal and non-diverse front.

In our literature review, we found the following research study that used DE to optimize the design of soft robots:

- Tan et al. [114] used DE to calibrate a soft robot hand, specifically for the kinematics of the finger and the relationship between fingers. They compared DE with other two non-EC techniques: IPA (see Section 3.2.1) and LMA (see Section 3.2.2), claiming that these two methods outperformed DE in terms of identification accuracy and processing time.

3.1.4. Estimation of distribution algorithm (EDA)

The EDA [84] is an EC technique exploiting probabilistic models in the search process. Unlike traditional EC methods that rely on crossover and mutation to create offspring, EDAs rely on the concept of building and refining probability distribution that characterizes the relationships between variables in the search space [125]. This distribution is then used to generate candidate solutions – that is, the distribution of high-performing solutions guides the exploration of the search space, in a sort of *knowledge transfer* within individuals in the population. Thanks to this process, EDA can efficiently locate optimal solutions and converge with few iterations.

In our literature review, we found the following research study that used EDA to optimize the design of soft robots:

- Cheong et al. [113] used EDA to optimize the design and kinematic calculation of a multi-joint continuum robot by minimizing the total actuator torque applied at the base of each continuum joint.

3.1.5. Evolution strategy (ES) and covariance matrix adaptation evolution strategy (CMA-ES)

ESs [126, 127] were introduced specifically for design optimization problems [126, 128, 129]. Their working principle is similar to that of a GA without the crossover operator, where new solutions are sampled based on a multivariate normal distribution, and pairwise dependencies between variables are represented by a covariance matrix. The CMA-ES [85, 86] is a method to update ES's covariance matrix, which allows the algorithm to dynamically adjust the exploration and exploitation of the search space.

In our literature review, we found the following research studies that used CMA-ES to optimize the design of soft robots:

- Medvet et al. [90] used CMA-ES to optimize the morphology of a voxel-based soft robot by maximizing the distance it traveled during a locomotion task;
- Ferigo et al. [116] used CMA-ES to optimize the sensory apparatus (i.e., the type of sensors and number) of a voxel-based soft robot by maximizing its average speed during a locomotion task; and
- the same authors [19] published a different study using CMA-ES to optimize the location of sensors in a voxel-based soft robot by maximizing the intensity of their babbling (i.e., the phase of the robot's life when it explores its sensing ability).

3.1.6. Adaptive stochastic search (ASS)

ASS [93] is a method for solving non-differentiable optimization problems by converting them into a differentiable one on the parameter space of the parameterized sampling distribution; then, it uses a direct gradient search method to find fitter solutions – that is, it approximates the gradient of the objective function by evaluating random perturbations around some nominal value of the variable of optimization. Although this method does not directly fall within the area of EC, we categorized it as such due to its many similarities with EC (e.g., being population-based).

In our literature review, we found the following research study that used ASS to optimize the design of soft robots:

- Exarchos et al. [118] used ASS to optimize the design of a soft growing robot (i.e., solve the inverse kinematics) for a specific task by minimizing the error between the end effector and desired target, in tasks having multiple targets and obstacles.

3.1.7. NeuroEvolution

NeuroEvolution refers to strategies based on machine learning and neural networks [130] in which the weights and topology of the network are evolved by EAs.

3.1.7.1. NeuroEvolution of augmenting topologies (NEAT). NNEAT [87] is an efficient neuro-evolutional method that (i) employs a principled method of crossover of different topologies, (ii) protects structural innovation using *speciation* (i.e., favorites good solutions in diverse regions of the search space for multi-optima retrieval [131]), and (iii) allows the network to incrementally grow from a minimal structure. When this algorithm is applied not only to optimize the network's weights but also its structure and the activation functions of its neurons, then the algorithm is defined as Compositional Pattern Producing Network using NeuroEvolution of Augmenting Topologies (CPPN-NEAT) [88]. This strategy is extremely convenient in deep learning [132], where the structure of the network presents many hidden layers, such that researchers do not need to fix the number of hidden layers in advance – which is usually a burden.

In our literature review, we found the following research studies that used NEAT/CPPN-NEAT to optimize the design of soft robots:

- Kimura et al. [50] used NEAT [87] for designing voxel-based soft robots, which is based on EAs for tuning the weights of the network. Specifically, after generating a population of designs with NEAT, these designs are used as the initial population of SPEA2 (see Section 3.1.1) to satisfy multiple environmental conditions (i.e., terrestrial and aquatic locomotion). This is defined as multi-objective NEAT (MO-NEAT) [89]. However, the authors claimed that SPEA2 was not sufficient to retrieve a diverse Pareto front, so they modified it by adding the crowding distance of NSGA-II (i.e., the niching operator that guarantees diversity in the Pareto set – see Section 3.1.1).
- Cheney et al. [6] used CPPN-NEAT and MO-NEAT to optimize the topology of voxel-based soft robots to maximize (i) the amount of stretch of the robot during reaching tasks and (ii) the size of the robot while performing locomotion. Specifically, they claim that their implementation does not require the removal of speciation but simply performs a Pareto ranking of individuals within each species (similarly to NSGA-II – see Section 3.1.1).

In the context of soft-robot optimization, we found other two studies using NEAT and CPPN-NEAT, integrated with two additional techniques:

- *Speciated Evolver* (SE), an EA proposed for and implemented directly in soft-robot optimization by Medvet et al. [90], which is a variant of NEAT using speciation to protect innovations introduced by modifications in the topology rather than optimizing the topology itself; the authors used it to optimize the morphology of a voxel-based soft robot by maximizing the distance it traveled during a locomotion task; and

Table VI. *Non-evolutionary computation techniques.*

Abbreviation	Algorithm
IPA	Interior-Point Algorithm [133]
LMA	Levenberg–Marquardt Algorithm [134, 135]
NMSM	Nelder–Mead Simplex Method [136]
CCD	Cyclic Coordinate Descent [137]
GRD	Greedy Algorithm [138]
FABRIK	Forward And Backward Reaching Inverse Kinematics [139]
PS	Pattern Search [140]
COBYLA	Constrained Optimization by Linear Approximations [141]
BO	Bayesian Optimization [142]
HC	Hill Climbing [102]
SA	Simulated Annealing [143]
qpOASES	Quadratic Programming Online Active Set Strategy [144]
SDM	Steepest Descent Method [145]
MMA	Method of Moving Asymptotes [146]
NRM	Newton–Raphson Method [147]
BFGSA	Broyden–Fletcher–Goldfarb–Shanno Algorithm [148]

- *Novelty Search* (NS) [91], which evaluates solutions with respect to their *novelty* rather than the objective function. The novelty is defined as in the objective space rather than in the search space: for example, a solution (i.e., a robot) exhibiting a different trajectory from any previous solution is considered as *novel*. Methenitis et al. [117] used NEAT with NS to optimize both the morphology and locomotion strategy of voxel-based soft robots by defining their behaviors (i.e., trajectories and pace, number of voxels touching the ground, kinetic energy, and pressure) and maximizing the distances among them and guaranteeing diversity.

3.1.7.2. Gaussian process regression (GPR) tuned with a GA. Gaussian process regression [101] is a method used in machine learning that computes the variance in the estimate of the objective function in a population of random variables having (consistent) joint Gaussian distributions. Although it does not classify as an EC technique, in the context of our literature review we included it within neuroEvolution methods as one study used a GA (see Section 3.1.1) to tune GPR for soft-robot design:

- Fathurrohim et al. [100] used EC to tune the hyperparameters of GPR; specifically, they used a hybrid technique combining a GA and the hill climbing algorithm (see Section 3.2.11). GPR was used for optimizing the stiffness of a soft-robot fish's fin by maximizing the time-averaged thrust force it produces.

3.2. Soft robot designs with other optimization techniques (non-evolutionary)

Table VI reports the list of non-EC techniques used in the studies we collected, along with their abbreviation and a reference to the original article where the method was proposed. Table 7 shows the list of studies we collected, in which non-EC techniques were used to optimize kinematics, topology, locomotion, sensors, and energy. We observed that the most commonly used non-EC techniques are the interior-point algorithm, simplex methods such as Nelder–Mead, and many heuristic-based algorithms.

Table VII. *Designs with non-evolutionary computation techniques.*

Authors	**Type of robot**	**Application**	**Metrics**	**Optimization method**
Lloyd et al. [149]	Con, Mat	MSP, M	K	IPA [133]
Kim et al. [59]	Con	ELT	K, S	IPA [133]
Ros et al. [38]	Cab, Con	MSP, M	K	IPA [133]
Usevitch et al. [150]	Flu	ELT	T	IPA [133]
Tan et al. [114]	Cab, Con, Gri	M	K	IPA [133, 151], LMA [134, 135]
Lai et al. [152]	Cab, Con	M	K	LMA [134, 135]
Abbaszadeh et al. [31]	Bio, Mat	ELT	S, K	NMSM [136]
Burgner et al. [153]	Con	MSP	T, K	NMSM [136]
Bergeles et al. [154]	Con	MSP	T	NMSM [136]
Rucker et al. [155, 156]	Con	M, MSP	K	NMSM [136]
Rucker and Webster [157]	Cab, Con	M	K	NMSM [136]
Zhang et al. [158]	Con	M	K	CCD [137]
Wu et al. [159, 160]	Con	ELT	K	FABRIK [139]
Koehler et al. [10]	Flu	M	T	GRD [138]
Bedell et al. [161]	Con	MSP	T	PS [140]
Anor et al. [162]	Con	MSP	K	PS [140]
Schiller et al. [35]	Bio, Flu	ELT	K, L, E	COBYLA [141]
Adagolodjo et al. [163]	Cab, Gri	M	K	qpOASES [144]
Coevoet et al. [164]	Cab	MSP	K	qpOASES [144]
Coevoet et al. [165]	Man	MSP	K	qpOASES [144]
Fathurrohim et al. [100]	Bio	ELT	F	GPR [101] with GA [77, 78] + HC [102]
Ghoreishi et al. [166]	Con, Flu	M, MSP	T	BO [142], IPA [133], SA [143]
Chen et al. [167, 168]	Flu, Gri	M	T	SDM [145]
Wang et al. [169]	Cab, Gri	M	T	MMA [146]
Li et al. [41]	Gri	M	T	MMA [146]
Morgan et al. [170]	Gri	M	E	NRM [147]
Maloisel et al. [171]	Man	M, ELT	T	BFGSA [172]

3.2.1. Interior-point algorithm (IPA)

IPAs [133, 151] are mostly used for convex optimization problems [133]. They are iterative methods that retrieve the optimal solution by traversing the interior of the feasible search space (i.e., within the problem's constraints). Remarkably, IPAs can solve linear programming problems in polynomial time [151]. However, their gradient-based nature requires the objective function to be differentiable – implementations for non-differentiable functions can also be found in the literature [173]). There are even readily available software packages for nonexpert users of these methods: for example, MATLAB Optimization Toolbox offers an implementation of IPA with the commands `fmincon`, `GlobalSearch`, and `MultiStart`.

In our literature review, we found the following research studies that used IPA to optimize the design of soft robots:

- Tan et al. [114] used IPA in a study for the calibration of a soft robot hand, specifically for the kinematics of the finger and the relationship between fingers. They compared IPA with other

two techniques: LMA (see Section 3.2.2) and DE (see Section 3.1.3), claiming that both IPA and LMA outperformed DE – the only EC technique – in terms of identification accuracy and processing time;
- Lloyd et al. [149] used IPA to optimize the magnetic field of a soft-continuum-magnetic robot such that it conforms to a desired shape by minimizing contact with the environment;
- Kim et al. [59] used IPA to optimize the curvature sensor placement of a continuum robot by minimizing the shape and tip error between its reconstruction and mechanics-based model;
- Ros et al. [38] used multiple runs of IPA (MATLAB's `GlobalSearch`) to optimize the joint angle limits of a continuum robot by minimizing the distance between its tip and the possible associated motion plans (i.e., targets);
- Usevitch et al. [150] used IPA to optimize the shape of an isoperimetric soft robot by maximizing the volume it encloses; and
- Ghoreishi et al. [166] used IPA (i) to implement a hybrid method based on IPA and BO (see Section 3.2.7) for aligning a soft catheter robot to a human blood vessel and (ii) for comparison, showing that their method is more performant.

3.2.2. Levenberg–Marquardt algorithm (LMA)
The LMA [134, 135], or the damped least-squares method, is an iterative gradient-based procedure specifically used to solve nonlinear least-squares problems (i.e., curve fitting) [134, 135]. Being based on gradient descent, LMA cannot be applied to non-differentiable functions.

In our literature review, we found the following research studies that used LMA to optimize the design of soft robots:

- Tan et al. [114] used LMA in a study for the calibration of a soft robot hand, specifically for the kinematics of the finger and the relationship between fingers. They compared LMA with other two techniques: IPA (see Section 3.2.1) and DE (see Section 3.1.3), claiming that both LMA and IPA outperformed DE – the only EC technique – in terms of identification accuracy and processing time.
- Lai et al. [152] used LMA to optimize the trajectory of a continuum robot by minimizing the distance between its tip and the target (inverse-kinematics solver).

3.2.3. Steepest descent method
The steepest descent method (SDM) [145], also known as gradient descent, is a gradient-based method that iteratively moves the current solution in the direction of the local downhill gradient.

In our literature review, we found the following research studies that used SDM to optimize the design of soft robots:

- Chen et al. [167, 168] used SDM to optimize the skeleton topology of soft-pneumatic network grippers by maximizing the bending angle of the actuator while sustaining prescribed interacting forces.

3.2.4. Nelder–Mead simplex method (NMSM)
The simplex algorithm is a method for linear programming (i.e., a linear objective function subject to linear constraints) [174]. It iteratively moves from one feasible solution to another along the edges of a *polytope* (i.e., a convex region defined by the constraints), selects a pivot element, and performs operations to improve the objective function value until an optimal solution is found. It is a gradient-free method.

While the classic simplex algorithm was designed for linear programming, the NMSM [175] variant can effectively optimize nonlinear functions, as well as non-differentiable problems or problems with discontinuities in the search space. Its main advantages are its independence from the gradient information and its significant improvement of solutions in the first few generations. NMSM is also considered to be a heuristic search because it relies on exploration and adjustment strategies rather than explicit mathematical models to optimize functions. MATLAB Optimization Toolbox offers an implementation of NMSM with the command `fminsearch`.

In our literature review, we found the following research studies that used NMSM to optimize the design of soft robots:

- Abbaszadeh et al. [31] used the NMSM to find a set of strain gauge parameters of a soft-aquatic robot that would result in the most accurate kinematic calculation (i.e., minimum error between the kinematic computation and the optical sensor measurements);
- Burgner et al. [153] used the NMSM to optimize the volume design of a concentric-tube robot by maximizing the coverage of the required workspace for surgical operations;
- Bergeles et al. [154] used the NMSM to optimize the design of a concentric-tube robot by minimizing its length and curvature that can still reach all required paths in a stable manner (i.e., its topology), preferring this algorithm to the PS (see Section 3.2.13) they used in their previous study [161]; and
- Rucker et al. [155–157] used NMSM to optimize the design of a continuum robot by minimizing the Euclidean distances between the model prediction and experimental data from the perspective of its kinematics, statics, and dynamics.

3.2.5. Constrained optimization by linear approximations (COBYLA)

The COBYLA [141] is an algorithm for nonlinear constrained optimization. It is gradient-free, and it works by iteratively adjusting the values of decision variables using linear approximations of the objective and constraint functions. COBYLA is particularly efficient with noisy and discontinuous objective functions.

In our literature review, we found the following research study that used COBYLA to optimize the design of soft robots:

- Schiller et al. [35] used COBYLA to estimate the forward kinematics of a gecko-inspired soft robot by minimizing the energy while simulating its next pose during locomotion.

3.2.6. Cyclic coordinate descent (CCD)

CCD [137] is a deterministic algorithm that iteratively updates one decision variable (a *coordinate* in the search space) at a time while holding the others fixed, cycling through the variables until convergence. CCD can be applied to both minimization and maximization problems and is gradient-free and computationally efficient in problems with easily separable variables.

In our literature review, we found the following research study that used CCD to optimize the design of soft robots:

- Zhang et al. [158] used CCD to solve the inverse kinematics of a soft-continuum manipulator by minimizing the distance between the current position of the robot's end effector and the target position by moving the robotic joints in turn.

3.2.7. Bayesian optimization

Bayesian optimization (BO) [142] is a probabilistic and gradient-free optimization method. By exploiting Gaussian processes, it can retrieve optimal solutions without explicit knowledge of the objective function; and as such, it is considered a heuristic search. BO is particularly effective in scenarios where the objective function is complex, costly to evaluate, and lacks a known mathematical form.

In our literature review, we found the following research studies that used BO to optimize the design of soft robots:

- Ghoreishi et al. [166] designed a soft catheter for medical applications, which requires finding the best structure that aligns with the vessel shape investigated by the catheter. They used (i) a modified Bayesian technique to seek the optimal geometric and material properties of the soft catheter by minimizing the deviance of the actuated catheter from a desired vessel shape and (ii) IPA (see Section 3.2.1) to optimize the actuator moments for the specific vessel shape.

3.2.8. Method of moving asymptotes (MMA)

The MMA [146] is an algorithm for nonlinear constrained optimization – specifically designed for problems with a large number of constraints. The heuristic consists of iteratively keeping track of and adjusting the parameters defining lines and surfaces that approximate the problem's constraints (the *asymptotes*). This strategy allows MMA to converge to the optimal solution. MMA is considered a gradient-based optimization method as it involves derivatives in the constraint-handling process. It is claimed to be the most used optimization method for topology optimization [169].

In our literature review, we found the following research studies that used MMA to optimize the design of soft robots:

- Wang et al. [169] used MMA to optimize the topology of a soft gripper by minimizing a weighted linear combination of output displacements given external forces; and
- Li et al. [41] used MMA to optimize the topology of a soft gripper by minimizing the out-of-plane displacement while also maximizing the in-plane bending deformation of the soft finger under internal pressure and external loads.

3.2.9. Newton–Raphson method (NRM)

The NRM [147] is a numerical optimization algorithm used for problems with real-valued decision variables. It is gradient-based as it updates the current solution by taking steps proportional to the inverse of the objective function's second derivative to iteratively converge to the optimum.

In our literature review, we found the following research study that used NRM to optimize the design of soft robots:

- Morgan et al. [170] used NRM to optimize the design of a soft gripper by minimizing its total energy such that the spring lengths match the required features of the device and facilitate open-loop control.

3.2.10. Greedy algorithms (GRD)

GRD algorithms [138] are a class of optimization methods that make iterative choices only based on a defined heuristic. The term *greedy* refers to their approach of immediately selecting the best available option based on the current state, without considering the potential long-term consequences. While not always guaranteed to find the globally optimal solution, GRD algorithms are computationally efficient and straightforward. For example, they are efficient in path-finding problems where agents (or, in our context, robots) explore a free environment with no obstacles, as they directly move toward the target;

however, when obstacles are in the environment, they might not consider them in the heuristic and get stuck while their target is right behind the obstacle.

In our literature review, we found the following research study that used GRD algorithms to optimize the design of soft robots:

- Koehler et al. [10] implemented a GRD algorithm while optimizing the design of a soft 3D haptic shape display, specifically for actuator placement. The algorithm adds actuators iteratively to correct the worst error, which is calculated through control simulation.

3.2.11. Hill climbing (HC)

HC is a local search method that follows the objective function's direction toward better solutions. In brief, it is the gradient-free equivalent of gradient descend methods (see Section 3.2.3). It explores the search space starting from a random point and then deterministically updates its value by following the most fitting direction of the objective function. This heuristic makes HC very efficient in converging to an optimal solution; however, if the objective function features a non-smooth landscape (i.e., it has many local optima), the algorithm gets stuck in a local optima without the guarantee of finding the global one. Multiple runs of HC might be needed to converge to the global optima.

In our literature review, we found the following research study that used HC to optimize the design of soft robots:

- Fathurrohim et al. [100] combined HC and a GA (see Section 3.1.1) to tune the hyperparameters of GPR (see Section 3.1.7). This hybrid strategy applied to neuroEvolution was used for optimizing the stiffness of a soft-robot fish's fin by maximizing the time-averaged thrust force it produces.

3.2.12. Simulated annealing (SA)

SA [143] is a stochastic local search method for optimization. Its heuristic is inspired by the annealing process in metallurgy, in which the physical and chemical properties of a material are altered by reducing the temperature to increase its ductility (e.g., the process used to manufacture sword blades). Like HC (see Section 3.2.11), SA starts with an initial solution and explores the search space by accepting moves with a probability based on the Boltzmann distribution – that is, it can also accept less fitting solutions. If the next point has a better value, then SA accepts it and reiterates the procedure; if not, SA accepts it with a probability depending on how much worse the point is with respect to the previously generated one and how long the algorithm has been running to simulate *temperature* (i.e., hot at the beginning and cold toward the end). This allows SA to escape local optima, which is the main downside of HC. However, the cooling simulation should be very slow to enforce regularities of the objective's layout, resulting in long runs.

In our literature review, we found the following research study that used SA to optimize the design of soft robots:

- Ghoreishi et al. [166] used SA for comparison to evaluate their method based on BO (see Section 3.2.7) and IPA (see Section 3.2.1), showing that their method is more suitable for aligning a soft catheter robot to a human blood vessel.

3.2.13. Pattern search (PS)

PS [140] is a gradient-free algorithm that explores the search space by modifying the values of decision variables with different patterns. It is considered a heuristic search due to its systematic exploration strategies for exploring the search space, making it well-suited for complex optimization problems having, for example, discontinuous objective functions. MATLAB Optimization Toolbox offers an implementation of PS with the command `patternsearch`.

In our literature review, we found the following research studies that used PS to optimize the design of soft robots:

- Bedell et al. [161] used PS to optimize the design of a concentric-tube robot by minimizing its curvature and length for navigation of cluttered environments (specifically human arteries and heart). However, in a follow-up study on the same robot and with the same optimization criteria [154], the authors defined PS as "computationally inefficient due to its sampling approach" and decided to repeat their optimization using NMSM (see Section 3.2.4).
- Anor et al. [162] used PS to optimize the design of a continuum robot by minimizing its complexity (i.e., solved the inverse kinematics retrieving the minimal number of sections composing the robot) in neurosurgical procedures (i.e., specifically designed to work inside the human brain). They minimized (i) the percentage of the robot residing outside the workspace and (ii) the total length of the robot to avoid unnecessary looping or coiling.

3.2.14. Forward and backward reaching inverse kinematics (FABRIK)

FABRIK [139] is a gradient-free heuristic algorithm commonly used in computer graphics and robotics to solve inverse kinematics problems. It iteratively adjusts the values of joint angles in a kinematic chain until the end effector reaches the desired position; specifically, it alternates a *forward pass* in which joints are updated from the base to the end effector and a *backward pass* in which joints are updated from the end effector to the base.

In our literature review, we found the following research studies that used (or rather, were inspired by) FABRIK to optimize the design of soft robots:

- Wu et al. [159, 160] introduced a heuristic algorithm specifically for solving the inverse kinematics of continuum robots, used for path planning and obstacle avoidance. They named it the Continuum Robot Reaching Inverse Kinematics (CRRIK), and it is based on FABRIK while inspired by the physical process of *pulling a rope with a fixed end*. This algorithm features a high convergence rate and low computational cost, which are suitable for real-time applications, and it is the most efficient when compared with other inverse-kinematics solvers (i.e., FABRIK [139], CCD [137], Follow The Leader (FTL) [176], or calculating the Jacobian matrix).

3.2.15. Parametric quadratic programming algorithm (PQP)

PQP is a method to solve quadratic programming problems [177] that trace optimal solutions on a homotopy path between two quadratic problem instances – that is, when one continuous function can be continuously deformed into another, such that it is possible to move from one topological space to another [178]. This algorithm forms the basis for the algorithms included in the open-source C++ software package qpOASES [144], which outperforms other popular academic and commercial quadratic-programming solvers on many small-to medium-scale convex test examples. PQP and qpOAESIS are not gradient-based methods.

qpOASES is mostly used to optimize the design of soft robots by solving the inverse kinematics while minimizing the displacement of its actuators from the desired position. This was applied to:

- a cable-driven soft gripper by Adagolodjo et al. [163];
- a cable-driven soft robot by Coevoet et al. [164]; and
- a soft-rigid hybrid arm by Coevoet et al. [165] .

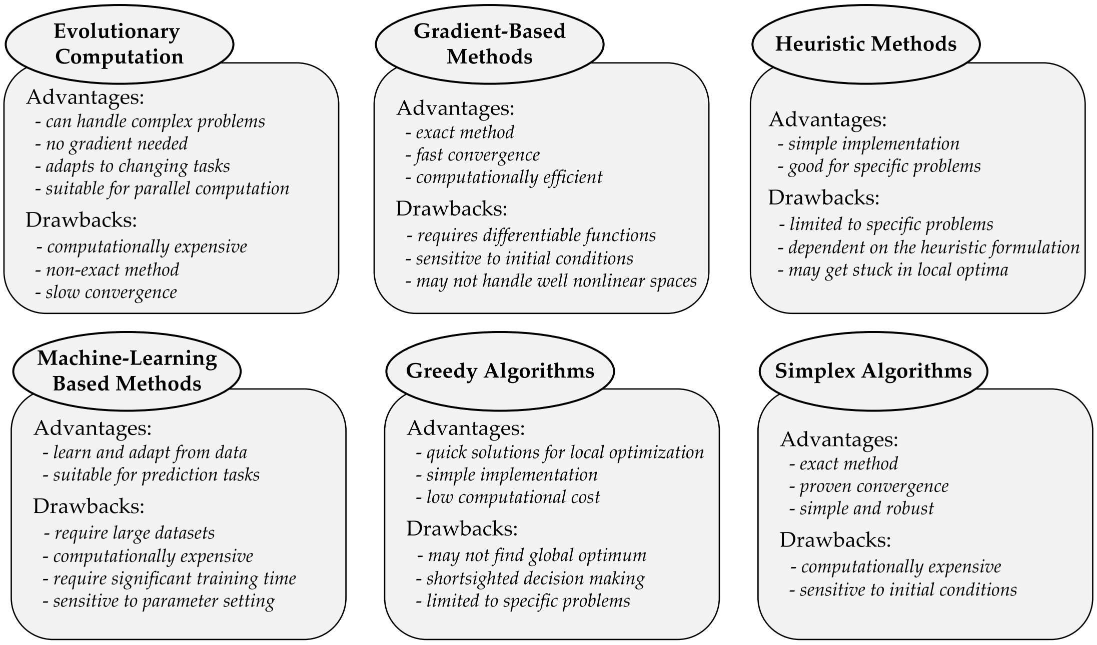
Evolutionary Computation
Advantages:
- can handle complex problems
- no gradient needed
- adapts to changing tasks
- suitable for parallel computation
Drawbacks:
- computationally expensive
- non-exact method
- slow convergence
Gradient-Based Methods
Advantages:
- exact method
- fast convergence
- computationally efficient
Drawbacks:
- requires differentiable functions
- sensitive to initial conditions
- may not handle well nonlinear spaces
Heuristic Methods
Advantages:
- simple implementation
- good for specific problems
Drawbacks:
- limited to specific problems
- dependent on the heuristic formulation
- may get stuck in local optima
Machine-Learning Based Methods
Advantages:
- learn and adapt from data
- suitable for prediction tasks
Drawbacks:
- require large datasets
- computationally expensive
- require significant training time
- sensitive to parameter setting
Greedy Algorithms
Advantages:
- quick solutions for local optimization
- simple implementation
- low computational cost
Drawbacks:
- may not find global optimum
- shortsighted decision making
- limited to specific problems
Simplex Algorithms
Advantages:
- exact method
- proven convergence
- simple and robust
Drawbacks:
- computationally expensive
- sensitive to initial conditions

Figure 5. *Advantages and drawbacks schema of the optimization techniques analyzed in the survey.*

Specifically, they defined their model as a constrained optimization problem and retrieved the optimal Lagrangian multipliers [179]. We observed that this method is usually associated with robots modeled with finite element method.

3.2.16. Broyden–Fletcher–Goldfarb–Shanno algorithm (BFGSA)

The BFGSA [148] is a gradient-based method for unconstrained optimization. It converges to the optimal solution by iteratively estimating the objective function's inverse Hessian matrix (a square matrix composed of second-order partial derivatives of the objective function).

In our literature review, we found the following research study that used BFGSA to optimize the design of soft robots:

- Maloisel et al. [171] used BFGSA to optimize the load-displacement profile and design parameters of a soft-flexible link mechanism by minimizing its force equilibrium equation.

4. Discussion

While conducting this survey, we identified several prevailing patterns, concerns, and obstacles. In this section, we will (i) summarize our findings on how optimization techniques are implemented and communicated in the field of soft robot design and (ii) offer guidance to future designers on enhancing the quality of their research and their valuable contributions to the community. Figure 5 summarizes the most common optimization techniques analyzed in this survey by listing advantages and drawbacks.

4.1. Importance of using mathematical optimization in engineering

Out of the 954 studies we retrieved during our survey, 165 did not include any mathematical optimization. Because they specified *optimization* as a future plan, we still retrieved them in our search and consequently discarded them (as shown in Figure 1). This is interesting because results obtained from prototypes and preliminary works can undergo massive differences and improvements thanks to mathematical optimization. While we acknowledge that it is not expected to complete a full-scale robot implementation in a single work, we invite roboticists to apply optimization techniques as the first stages of a soft robot design.

We believe that designers and roboticists may hesitate to implement complete optimization methods at the first design stage due to a potential need for interdisciplinary work with computer scientists, who would implement these algorithms. With this survey, we highlight the simple and user-friendly tools that are available for non-mathematicians and non-computer scientists to claim meaningful and comparable findings (see Section 4.3).

4.2. Importance of specifying the optimization method

While conducting this survey, our primary focus was on providing thorough descriptions of the parameters and methodologies employed in each study. However, we often encountered a significant challenge due to a lack of details in the authors' descriptions. This shortcoming led us to exclude 11 studies from our literature review (see Figure 1, in which the authors did not clearly specify their method). Some examples are the following:

- Camarillo et al. [180] only reported using a "linear programming solver" without specifying nor referencing any specific method.
- Chen et al. [11] only reported that the optimization algorithm undergoes an iterative process in MATLAB – without detailing whether it was a built-in script from the toolbox or self-coded;
- Gilbertson et al. [181] reported maximizing the extension of a soft robot per input pressure cycle (i.e., the distance it travels) by optimizing its mechanical properties (the orifice coefficients of the passive valves); however, it is unclear which methods they used, especially because they defined many optimization problems. To the best of our understanding, some problems were solved with NRM (see Section 3.2.9), whereas others with naive brute force. These ambiguities in the text led us to exclude this study from the review.
- Bern and Rus [37] and Fang et al. [182] used a *generic* gradient-based method to solve the robot's inverse kinematics, without specifying any details nor the name of the method; and
- Marchese and Rus [40] proposed a rigorous and detailed mathematical formulation to solve the inverse kinematics of a soft spacial fluid elastomer manipulator as an optimization problem; however, they did not specify which method they used.

For some studies included in the survey, the authors were somehow vague on the optimization method. For example:

- Dinakaran et al. [107] presented their own GA implementation (see Section 3.1.1) and referred to it as an "improved version"; however, it is unclear what the authors improved. The algorithm was described as a limited flowchart that did not fully detail genotype representation, genetic operator, or elitist methods. Furthermore, they presented no comparison with a classic GA.
- Hiller and Lipson [105] used the term EA without specifying which algorithm they used. We tracked back their references [183, 184] to conclude that they used a GA.
- Sui et al. [95] did not directly specify which EC technique they used, referring to them with a generic "evolution strategy." After some digging into the text, we found out that the CAD software they used (VoxCAD) has an integrated GA [185] add-on. Furthermore, using the term "evolution strategy" is inadvisable, since there is a specific EC technique named Evolution Strategy [186].

Lastly, we encountered similar issues regarding the formalization of the optimization problem. Ideally, this information should be explicitly articulated in an appropriate section with a full definition of the objective function(s), decision variables, and constraints (if any). It is essential to clarify whether the problem is minimized or maximized, such that readers can simply look for these specific keywords (i.e., "min," "max," "opt") to easily detect where the optimization is described. Unfortunately, authors often mention these important factors and decisions between the lines of their manuscripts, making their work hard to repeat. Examples included in our review are:

- Sui et al. [95] reported optimizing "soft robot deformation" but did not clearly define what it refers to in terms of the optimization problem to the best of our understanding – that is, did not define a proper objective function describing the practical implications on the robot; and
- Rieffel et al. [106], which we found hard to understand in terms of detecting the decision variables and even to discriminate fully whether this was design optimization, control optimization, or possibly both.

4.3. Methods available in MATLAB and other libraries

MATLAB is one of the most common tools for research, as it provides many libraries for different types of applications and fields. Specifically, MATLAB Optimization Toolbox offers the implementation of several methods reported in this survey: GA (`ga`), NSGA-II (`gamultiobj`), IPA (`fmincon`, `GlobalSearch`, and `MultiStart`), NMSM (`fminsearch`), and PS (`patternsearch`). Its popularity among researchers – including the authors we referenced in our survey – is due to its ease of use, robust documentation, and the efficiency of its algorithms. The latter alone justifies the authors' choice of picking one among these algorithms: roboticists might not be expert coders nor know every implementation detail of any optimization method; therefore, it is safe to use reliable methods offered by built-in libraries. We also found other libraries implementing optimization methods that are not from MATLAB, such as COBYLA (a Python library), qpOASES (a C++ library), and FABRIK (integrated with Unity3D in C# and Panda in Python).

Although using optimization methods as a black box is still a valid option, we would like to raise awareness of different algorithms such that roboticists can make an accurate and conscious choice on which method fits their needs. Based on our observations, many research groups had full knowledge of these optimization methods, which is evidenced by their ability to manipulate the implementations of these algorithms. For instance, Fathurrohim et al. [100] and Kimura et al. [59] combined different EAs to tune their neural networks, Wu et al. [159, 160] proposed an innovative method based on FABRIK (named CRRIK), Koehler et al. [10] designed their specific GRD algorithm, and Dinakaran et al. [107] claimed to have *improved* a GA (although they did not provide further details).

4.4. Claims on EC's advantages

While working on this survey, we observed that the literature has a similar amount of work for EC and non-EC methods. Although the choice of optimization method depends on various factors (including the specific problem at hand, available computational resources, and the expertise of the designer), many soft-robot studies relied on and praised the efficiency of EC. For example, Liu et al. [103] designed their soft gripper through simulation, by using an artificial neural network to estimate the bending force and the center of gravity. These estimated parameters were then successively optimized with EC to obtain the maximum force at the minimum offset of the design's center of gravity. The authors showed that this optimization reduced the trial and error design work and has great potential for effectively developing soft robots in industrial applications, thanks to EC. Medvet et al. [90] compared three different EAs (CMA-ES, GA, and SE) to optimize the morphology of their robot, showing that the algorithm plays a more important role than the way designs are encoded in the problem (i.e., representation of decision variables) in determining biodiversity. Specifically, they suggest that employing a diversity promotion mechanism based on a humanlike notion of species can result in better effectiveness, as well as in larger biodiversity. This is a characteristic specific to EC, as it pertains to the concept of exploitation and genotype.

Based on the relevant EC state of the art, supported by the evidence collected in this survey, we would like to summarize its advantages for optimizing the mechanical designs of soft robots:

- Soft robots often feature **non-differentiable and complex design spaces**, involving nonlinear relationships and interactions between multiple design variables. While many optimization methods rely on the assumption of differentiability, EC techniques do not suffer from this downside.
- Many of the studies analyzed in this survey feature **multi-objective search spaces**, either optimizing the same concept with size or path variations (e.g., finding the best combination of parameters to optimize locomotion on different terrains [50], different sizes [7, 20]) or optimizing completely independent concepts (e.g., finding the best combination of parameters to optimize dexterity and load-bearing capacity [99] or reaching multiple targets by minimizing the overall length, the amount of steering, and the error in reaching orientation [57, 118]). EC techniques are extremely efficient in finding optimal solutions in wide space(s) due to their population-based nature, contrary to classic methods that update a single solution at the time.
- Thanks to genetic operations such as selection, crossover, and mutation, EC techniques guarantee an exhaustive **exploration of the search space** while maintaining **diversity** in the population of solutions. This allows the algorithms to retrieve different and – potentially – better solutions. Mechanisms such as mutation and niching (especially for MOOP) also enforce diversity within the population, preventing premature convergence to suboptimal solutions and allowing the discovery of a wider range of trade-offs. For example, one of the reasons Gao et al. [112] reported choosing PSO (see Section 3.1.2) was for its "degree of parallelism and ability to obtain the optimal solution in many solutions," indicating the diversity in the converged population of solutions.
- EC encourages the exploration of **unconventional and innovative solutions** by introducing random variations. This allows EC techniques to discover novel designs that may not be immediately obvious to human designers or classical optimization methods – a characteristic that matches the needs of innovative designs such as soft robots. For example, many voxel-based soft robots [19, 50, 90, 96, 97, 104, 105, 116, 117] were also defined as *evolutionary robots* in the sense that they can evolve their morphology to perform specific tasks (usually locomotion tasks). Applying *evolutionary* algorithms on *evolutionary* robots is a perfect match, supported by the shared nomenclature.
- Due to the population-based nature of EC techniques, they are suitable for **parallel computation** to speed up runtime. Operations such as population initialization, fitness evaluation, and variation operators can be parallelized as they operate independently for each individual of the population.

4.5. Claims on EC's disadvantages

We acknowledge that the choice of optimization method depends on various factors, including the specific problem at hand, available computational resources, and the expertise of the designer. In addition, non-EC methods (or classical optimization methods) might be favorable and effective in specific situations, particularly when the design space is well-defined and the objectives are simple and easily differentiable. Some of these disadvantages are:

- Koehler et al. [10] claim that a GRD algorithm (see Section 3.2.10) is preferred to an EA because the objective function was derived from a **simulation-based optimization problem**. Since EAs work iteratively and stochastically, their time complexity (and therefore running time) is linear to the number of iterations and linearithmic to the size of the population; however, because the objective function must be evaluated for each solution in the population, this puts an additional toll on the runtime. The simulation used by Koehler et al. for their soft-haptic-shape display is computationally expensive due to the device's high number of DoFs; therefore, they preferred a GRD algorithm for faster convergence.
- Ghoreishi et al. [166] and Tan et al. [114] also share the concert on the **computational complexity** of EC methods. They suggest that the design of a robot – whether soft or not – is an offline process; therefore, the runtime of the optimizer can be negligible unless the optimizer takes years to converge.
- Booker et al. [187] provided an extensive analysis of the performance of optimizers based on the scale of problems, showing that EAs offer better performance than any other optimization or search method (i.e., EAs are *generically* more performant on any type of problem); however, they also showed that a **specialized algorithm** performs best on its specific problem. Although this might be the case for Koehler's soft-haptic-shape display, in which their GRD algorithm is considered as *the* specializer optimizer, EAs remain a valid alternative – as also shown in other studies on mechanical design for rigid exoskeletons [76] and one of the soft-robot study included in our survey [103].

4.6. Differences among EC techniques

In Section 3.1, we reported a brief walk-through on every EC technique employed in the studies included in the survey. These differences were also considered and reported by some authors while selecting the appropriate optimization method for their design:

- Abbaszadeh et al. [31] used three different optimization methods for their design: two EC techniques (GA and PSO) and NMSM (see Section 3.2.4). They reported no significant deviations in terms of performance of the retrieved design between these techniques.
- Cheong et al. [113] used both EDA (see Section 3.1.4) and GA for their continuum-robot design. They reported that EDA outperformed GAs with solutions $4\% - 15\%$ better in terms of optimality, as supported by the literature [188–192].
- Gao et al. [112] specifically choose PSO because, when compared with GA, the convergence speed is faster and employs many measures to avoid being trapped in a local optimum – although this can be overcome to some extent by increasing the amount of mutation in GA. They also reported the same claim for another EC technique, ant colony optimization (ACO) [193]; however, since they only referred to this from the state of the art without directly applying it to their design, we did not include ACO in Section 3.1.

4.7. When to use different EC techniques and which ones

In this survey paper, we observed that 29 studies relied on EC techniques, 30 studies relied on non-EC techniques, while 3 relied on both [31, 100, 114]). These numbers indicate that EC is quite popular among soft robot designers. We could observe that most of these authors made a conscious choice when selecting the appropriate optimization method to match their needs; however, we believe that such conscious choice is not the case for all research groups. While one of the advantages of EC techniques is that they can be applied to any optimization problem without being constrained to some specifics, several other aspects make EC suitable in engineering, with different algorithms featuring specific properties to be exploited. In this section, we offer a generalized guide on which type of algorithm to choose while facing a specific class of problems.

4.7.1. Multimodal optimization problems

Multimodal optimization problems refer to having multiple local or global optima (as detailed in Section 2), which often gets challenging through traditional methods. Finding the global optima in a non-smooth search space in addition to finding one or more alternative local optima requires additional computational efforts. Thanks to their population-based nature, EC techniques are great for dealing with such efforts, allowing them to retrieve multiple solutions in a single run without the need to restart the method.

However, evolving a population of solutions might still be insufficient to prevent being trapped in the same region of the search space. For example, the crossover operator of EAs promotes exploitation, which concentrates the search only on the region defined by the affected individuals. EC techniques then feature a suite of operators that facilitate the retrieval of more optima simultaneously: spatial segregation [194] or distribution [195] of a single population, partitioning [196], mating restrictions [197], elitism [198], sharing [199], niching [200], clearing [201], crowding [202], clustering [203], or by employing multi-objective optimization techniques [204]. A very efficient multimodal EA was recently proposed by Yenin et al. [205], named k-cluster Big Bang-Big Crunch algorithm (k-BBBC): inspired by the evolution of the universe, its ability to solve problems with more than 300 optima and up to32 decision variables makes it suitable for complex engineering problems.

In our survey, we observed that Gao et al. [112] chose PSO (see Section 3.1.2) as the optimization method due to its "degree of parallelism and ability to obtain the optimal solution in many solutions." This claim is somehow generic and does not directly explain what the authors were referring to; however, the best interpretation is that they refer to the diversity in the converged population of solutions and therefore the ability to retrieve multiple and diverse optima.

4.7.2. Multi-objective optimization problems

Population-based methods like EAs are particularly suitable for solving MOOPs as they can carry on more than one solution at a time. This property comes in handy in view of retrieving a set of trade-off solutions. However, they also become computationally expensive: (i) their runtime grows linearly with the number of objectives (even though this number becomes negligible when compared to the population size) and (ii) non-dominated sorting requires that every solution in the current population is compared against each other.

EC literature features a great number of MOEAs, of which the most known and popular appear to be NSGA-II [79] and SPEA2 [81]. While the differences between the two algorithms are mostly in terms of implementation and computational complexity [206], a clear ultimate winner between the two MOEAs cannot be determined [207], and both methods can be considered equivalent. When considering soft robot design, the complexity of such engineering problems makes the runtime play a decisive role (as also discussed in Section 4.5). With SPEA2 having a slightly higher computational complexity than NSGA-II, designers might need to take this observation into account. In this regard, Fitzgerald et al. [7, 20] used the latest version of NSGA (namely, NSGA-III [80]), which can solve problems with up to 15 objective functions with the same runtime as NSGA-II.

An interesting case is from Kimura et al. [50], which used SPEA2 to optimize NEAT (see Section 3.1.7). However, based on their claim, SPEA2 alone cannot distinguish between non-dominated individuals, so they had to integrate the crowding distance operator of NSGA-II in their implementation. We believe that these claims conflict with the state of the art, as both SPEA2 and NSGA-II were developed specifically to converge to a well-distributed and diverse Pareto set – possibly because the authors did not fully implement SPEA2. Specifically, they did not implement its *truncation* operator, which is responsible for guaranteeing diversity in the population of non-dominated solutions. We speculate that this might caused by two main reasons: (i) the description of the truncation operator from the original work of Zitzler et al. [81] is very hard to follow, and it is tempting for nonexpert readers to skip or miss that section; and (ii) Kimura et al. might have checked for strong dominance rather than weak dominance while assigning fitness to the individuals. Our speculation is based on the fact that the exact same problem happened to us while implementing SPEA2 the first time. Ultimately, we believe that a proper implementation of SPEA2 would be sufficient to optimize NEAT in the fashion proposed by Kimura et al.

Lastly, we report a direct comparison between two works included in this survey addressing the same problem: Exarchos et al. [118] and Stroppa [57] formalized a strategy to retrieve the optimal design of a soft growing manipulator (specifically, the robot proposed by Do et al. [39]) for a specific task. This problem features many objectives: retrieve the link lengths of a soft robot that (i) minimizes the distance between the end effector and all the targets, (ii) reaches the targets with a given orientation, (iii) minimizes the amount of undulation of the robots to avoid wavy configurations, (iv) minimizes the overall length of the robot, and (v) avoids obstacles in the environment. While Exarchos et al. based their strategy on a weighted sum [61], Stroppa proposed an innovative methodology named RP (see Section 3.1.1). Besides overcoming the many disadvantages of weighted-sum methods for MOOPs (e.g., the burden of choosing weights for each objective, not being able to solve problems featuring non-convex Pareto fronts, etc.), Stroppa showed that their method outperformed the one proposed by Exarchos et al. by providing shorter and less wavy robots. RP can be applied to any population-based method, and it is recommended for those problems in which priorities between objectives are recognizable.

5. Conclusion

Optimization strategies are commonly employed in developing intricate engineering structures like robotic systems. Given the adaptable nature of soft robotics, employing sophisticated optimization techniques is crucial for achieving effective results. This article reviews the current literature concerning optimization techniques being used in the design and realization of soft robotic systems. Throughout the paper, the term “soft” is defined from a broader perspective referring to any system making use of non-rigid materials such as fabric, elastomers (like silicone rubber), hydrogels, flexible polymers, shape memory alloys, organic substances, or a blend of synthetic and biological elements. The survey in the paper is structured around:

- the type of robot (bio-inspired, continuum, gripper, material-property actuated);
- the field of application (exploration/locomotion/tracking, grasping/manipulation, or medical surgery/medical procedures);
- the optimization metrics (topology, kinematics, force, locomotion, sensor placement, energy consumption); and
- the optimization method (categorized under evolutionary and non-EC techniques).

The presented review keeps a particular focus on the EC techniques that are widely utilized in engineering optimization. Therefore, this study also illustrates the prevalent use of EC in the domain of soft robot design, highlighting their appealing characteristics – such as (i) concurrent evaluation of multiple solutions (population-based), (ii) independence from gradient information, (iii) convergence, and (iv) versatility in tackling diverse optimization problems including those with multiple objectives and many local optima. Additionally, we prepared a comprehensive discussion detailing our observations on optimization method implementation and offering recommendations for future robot designers. We encourage designers to embrace optimization techniques early in the design process, leveraging the available simple and user-friendly tools for meaningful and comparable results. We invite future authors to consider these recommendations for better clarity and accessibility to readers.

In addition to the advancements presented in this study, there are several promising directions for future research in soft robot design and tracking: (i) enhancing computational efficiency in optimization algorithms, which could significantly reduce the time and resources required for robot design – especially with EC; (ii) employing machine learning techniques thanks to their capability to respond to dynamic environmental conditions and task requirements; (iii) investigating new materials and fabrication methods to produce soft robots with improved performance, durability, and functionality; and (iv) promoting interdisciplinary collaborations for fostering innovative solutions, particularly with materials science, biology, and – as shown in this survey – artificial intelligence. These potential research directions can address current limitations and open new possibilities for developing more advanced and capable soft robotic systems.

Author contributions. Investigation, F.J.M. and J.B.; data curation, F.S., F.J.M., J.B., and M.S.; visualization, F.S. and F.J.M.; writing – original draft preparation, F.S., F.J.M., and J.B.; writing – review and editing, F.S., M.S., and E.B.; supervision, F.S. and M.S. All authors have read and agreed to the published version of the manuscript.

Financial support. This work is funded by TUBİTAK within the scope of the 2232-B International Fellowship for Early Stage Researchers Program number 121C145.

Competing interests. The authors declare that they have no conflicts of interest regarding the publication of this paper.

Data availability statement. Not applicable.

Ethical standards. None.

References

[1] F. Chen and M. Y. Wang, “Design optimization of soft robots: A review of the state of the art,” *IEEE Robot Autom Mag* **27**(4), 27–43 (2020).
[2] M. Pellicciari, A. Avotins, K. Bengtsson, G. Berselli, N. Bey, B. Lennartson and D. Meike, “Areus-Innovative Hardware and Software for Sustainable Industrial Robotics,” **In:** *2015 IEEE International Conference on Automation Science and Engineering (CASE)*, (IEEE, 2015) pp. 1325–1332.
[3] W. S. Barbosa, M. M. Gioia, V. G. Natividade, R. F. F. Wanderley, M. R. Chaves, F. C. Gouvea and F. M. Gonçalves, “Industry 4.0: Examples of the use of the robotic arm for digital manufacturing processes,” *Int J Interact Des Manuf* **14**(4), 1569–1575 (2020).
[4] G. Runge, J. Peters and A. Raatz, “Design Optimization of Soft Pneumatic Actuators using Genetic Algorithms,” **In:** *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, (IEEE, 2017) pp. 393–400.
[5] G. M. Whitesides, “Soft robotics,” *Angew Chem Int Ed* **57**(16), 4258–4273 (2018).
[6] N. Cheney, J. Bongard and H. Lipson, “Evolving Soft Robots in Tight Spaces,” **In:** *Proceedings of the 2015 annual conference on Genetic and Evolutionary Computation*, (2015) pp. 935–942.
[7] S. Fitzgerald, G. Delaney, D. Howard and F. Maire, “Evolving Polydisperse Soft Robotic Jamming Grippers,” **In:** *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, (2022) pp. 707–710.
[8] A. K. Hongying Zhang, F. Chen, J. Fuh and M. Wang, “Topology optimized multimaterial soft fingers for applications on grippers, rehabilitation, and artificial hands,” *IEEE/ASME Trans Mech* **24**(1), 120–131 (2018).
[9] J. Wang, Y. Fei and Z. Liu, “Fifobots: Foldable soft robots for flipping locomotion,” *Soft Robot* **6**(4), 532–559 (2019).
[10] M. Koehler, N. Usevitch and A. Okamura, “Model-based design of a soft 3-d haptic shape display,” *IEEE Trans Robot* **36**(3), 613–628 (2020).
[11] F. Chen, W. Xu, H. Zhang, Y. Wang, J. Cao, M. Wang, H. Ren, J. Zhu and Y. F. Zhang, “Topology optimized design, fabrication, and characterization of a soft cable-driven gripper,” *IEEE Robot Autom Lett* **3**(3), 2463–2470 (2018).
[12] G. B. Dantzig, “Linear programming,” *Oper Res* **50**(1), 42–47 (2002).
[13] L. Wolsey. *Integer Programming* (John Wiley & Sons, Hoboken, NJ, 2020).
[14] K. Schittkowski, C. Zillober and R. Zotemantel, “Numerical comparison of nonlinear programming algorithms for structural optimization,” *Struct Optim* **7**(1-2), 1–19 (1994).
[15] J. Nocedal and S. Wright, “Quadratic Programming,” **In:** *Numerical Optimization* (Springer, 2006) pp. 448–492.
[16] P. Kall, S. Wallace and P. Kall, *Stochastic Programming* (Princeton University Press, Princeton, NJ, 1994).
[17] R. Bellman, *Dynamic Programming*, vol. 5 (Springer, Berlin, Germany, 1994).
[18] T. Back and H.-P. Schwefel, “Evolutionary Computation: An Overview,” **In:** *Proceedings of IEEE International Conference on Evolutionary Computation*, (IEEE, 1996) pp. 20–29.

[19] A. Ferigo, E. Medvet and G. Iacca, "Optimizing the sensory apparatus of voxel-based soft robots through evolution and babbling," *SN Comp Sci* **3**(2), 1–17 (2022).

[20] S. Fitzgerald, G. Delaney, D. Howard and F. Maire, "Evolving Soft Robotic Jamming Grippers," **In:** *Proceedings of the Genetic and Evolutionary Computation Conference*, (2021) pp. 102–110.

[21] E. Zardini, D. Zappetti, D. Zambrano, G. Iacca and D. Floreano, "Seeking Quality Diversity in Evolutionary Co-Design of Morphology and Control of Soft Tensegrity Modular Robots," **In:** *Proceedings of the Genetic and Evolutionary Computation Conference*, (2021) pp. 189–197.

[22] F. Stroppa, A. Soylemez, H. T. Yuksel, B. Akbas and M. Sarac, "Optimizing exoskeleton design with evolutionary computation: An intensive survey," *Robotics* **12**(4), 106 (2023).

[23] L. Schiller, A. Seibel and J. Schlattmann, "Toward a gecko-inspired, climbing soft robot," *Front Neurorobot* **13**, 106 (2019).

[24] S. M. Youssef, M. A. Soliman, M. A. Saleh, A. H. Elsayed and A. G. Radwan, "Design and control of soft biomimetic pangasius fish robot using fin ray effect and reinforcement learning," *Sci Rep* **12**(1), 21861 (2022).

[25] E. Medvet, A. Bartoli, A. De Lorenzo and S. Seriani, "2d-vsr-sim: A simulation tool for the optimization of 2-d voxel-based soft robots," *SoftwareX* **12**, 100573 (2020).

[26] Evolutionary Synthesis of Sensing Controllers for Voxel-based Soft Robots, ALIFE 2019: The 2019 Conference on Artificial Life of ALIFE 2023: Ghost in the Machine: Proceedings of the 2023 Artificial Life Conference, **7**(2019).

[27] G. Phanomchoeng, P. Pitchayawetwongsa, N. Boonchumanee, S. Lin and R. Chancharoen, "Grasping profile control of a soft pneumatic robotic gripper for delicate gripping," *Robotics* **12**(4), 107 (2023).

[28] K. Chen, T. Li, T. Yan, F. Xie, Q. Feng, Q. Zhu and C. Zhao, "A soft gripper design for apple harvesting with force feedback and fruit slip detection," *Agriculture* **12**(11), 1802 (2022).

[29] C. F. R. Costa and J. C. P. Reis, "End-point position estimation of a soft continuum manipulator using embedded linear magnetic encoders," *Sensors* **23**(3), 1647 (2023).

[30] N. Van Pho, S. B. Dhyan, V. Mai, B. S. Han and W. T. Chow, "Bioinspiration and biomimetic art in robotic grippers," *Micromachines* **14**(9), 1772 (2023).

[31] S. Abbaszadeh, R. Leidhold and S. Hoerner, "A design concept and kinematic model for a soft aquatic robot with complex bio-mimicking motion," *J Bionic Eng* **19**(1), 16–28 (2022).

[32] M. Tesch, J. Schneider and H. Choset, "Expensive Multiobjective Optimization for Robotics," **In:** *2013 IEEE International Conference on Robotics and Automation*, (IEEE, 2013) pp. 973–980.

[33] C. Tang, H. Huang and B. Li, "Design and Control of a Magnetic Driven Worm-Like Micro-Robot," **In:** *2021 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, (IEEE, 2021) pp. 1304–1308.

[34] M. Pollayil, C. Della, G. Mesesan, J. Englsberger, D. Seidel, M. Garabini, C. Ott, A. Bicchi and A. Albu-Schaffer, "Planning Natural Locomotion for Articulated Soft Quadrupeds," **In:** *2022 International Conference on Robotics and Automation (ICRA)*, (IEEE, 2022) pp. 6593–6599.

[35] L. Schiller, A. Seibel and J. Schlattmann, "A lightweight simulation model for soft robot's locomotion and its application to trajectory optimization," *IEEE Robot Autom Lett* **5**(2), 1199–1206 (2020).

[36] C. Ahn, X. Liang and S. Cai, "Bioinspired design of light-powered crawling, squeezing, and jumping untethered soft robot," *Adv Mater Technol* **4**(7), 1900185 (2019).

[37] J. Bern and D. Rus, "Soft ik with stiffness control," **In:** *2021 IEEE 4th International Conference on Soft Robotics (RoboSoft)*, (IEEE, 2021) pp. 465–471.

[38] L. Ros-Freixedes, A. Gao, N. Liu, M. Shen and G.-Z. Yang, "Design optimization of a contact-aided continuum robot for endobronchial interventions based on anatomical constraints," *Int J Comput Ass Rad* **14**(7), 1137–1146 (2019).

[39] B. H. Do, V. Banashek and A. M. Okamura, "Dynamically Reconfigurable Discrete Distributed Stiffness for Inflated Beam Robots," **In:** *International Conference on Robotics and Automation*, (IEEE, 2020) pp. 9050–9056.

[40] A. D. Marchese and D. Rus, "Design, kinematics, and control of a soft spatial fluidic elastomer manipulator," *Int J Robot Res* **35**(7), 840–869 (2016).

[41] D. C. Li, S. T. Chen, Z. N. Song, J. L. Liang, X. Y. Zhu and F. F. Chen, "Tailoring the in-plane and out-of-plane stiffness of soft fingers by endoskeleton topology optimization for stable grasping," *Sci China Technol Sci* **66**(11), 1–10 (2023).

[42] F. Stroppa, M. Selvaggio, N. Agharese, M. Luo, L. H. Blumenschein, E. W. Hawkes and A. M. Okamura, "Shared-control teleoperation paradigms on a soft-growing robot manipulator," *J Intell Robot Syst* **109**(2), 30 (2023).

[43] M. M. Coad, L. H. Blumenschein, S. Cutler, J. A. R. Zepeda, N. D. Naclerio, H. El-Hussieny, U. Mehmood, J.-H. Ryu, E. W. Hawkes and A. M. Okamura, "Vine robots: Design, teleoperation, and deployment for navigation and exploration," *IEEE Robot Autom Mag* **27**(3), 120–132 (2019).

[44] W. Thongking, A. Wiranata, A. Minaminosono, Z. Mao and S. Maeda, "Soft robotic gripper based on multi-layers of dielectric elastomer actuators," *J Robot Mech* **33**(4), 968–974 (2021).

[45] O. G. Akcan, K. Erdil, D. Korkut, E. A. Baran and Y. D. Gokdel, "Based piezoresistive force encoder for soft robotic applications," *IEEE Sens J* **22**(14), 13999–14007 (2022).

[46] C. Laschi, M. Cianchetti, B. Mazzolai, L. Margheri, M. Follador and P. Dario, "Soft robot arm inspired by the octopus," *Adv Robotics* **26**(7), 709–727 (2012).

[47] M. G. B. Atia, A. Mohammad, A. Gameros, D. Axinte and I. Wright, "Reconfigurable soft robots by building blocks," *Adv Sci* **9**(33), 2203217 (2022).

[48] G. Shin, Y. J. Choi, B. J. Jeon, I. Choi, S. Song and Y.-L. Park, "Soft electromagnetic artificial muscles using high-density liquid-metal solenoid coils and bistable stretchable magnetic housings," *Adv Funct Mater* 2302895 (2023).

[49] F. Pigozzi, Y. Tang, E. Medvet and D. Ha, "Evolving modular soft robots without explicit inter-module communication using local self-attention" (2022). arXiv preprint arXiv: 2204.06481.

[50] T. Kimura, R. Niiyama and Y. Kuniyoshi, “Modularized Genotype Combination to Design Multiobjective Soft-Bodied Robots,” **In:** *2021 IEEE 4th International Conference on Soft Robotics (RoboSoft)*, (IEEE, 2021) pp. 295–301.

[51] C. S. X. Ng, M. W. M. Tan, C. Xu, Z. Yang, P. S. Lee and G. Z. Lum, “Locomotion of miniature soft robots,” *Adv Mater* **33**(19), 2003558 (2021).

[52] F. Stroppa, M. Luo, K. Yoshida, M. M. Coad, L. H. Blumenschein and A. M. Okamura, “Human Interface for Teleoperated Object Manipulation with a Soft Growing Robot,” **In:** *International Conference on Robotics and Automation*, (IEEE, 2020) pp. 726–732.

[53] C. Choi, W. Schwarting, J. DelPreto and D. Rus, “Learning object grasping for soft robot hands,” *IEEE Robot Autom Lett* **3**(3), 2370–2377 (2018).

[54] M. M. Coad, R. P. Thomasson, L. H. Blumenschein, N. S. Usevitch, E. W. Hawkes and A. M. Okamura, “Retraction of soft growing robots without buckling,” *IEEE Robot Autom Lett* **5**(2), 2115–2122 (2020).

[55] G. Gerboni, J. D. Greer, P. F. Laeseke, G. L. Hwang and A. M. Okamura, “Highly articulated robotic needle achieves distributed ablation of liver tissue,” *IEEE Robot Autom Lett* **2**(3), 1367–1374 (2017).

[56] S. K. Talas, B. A. Baydere, T. Altinsoy, C. Tutcu and E. Samur, “Design and development of a growing pneumatic soft robot,” *Soft Robot* **7**(4), 521–533 (2020).

[57] F. Stroppa, “Design optimizer for planar soft-growing robot manipulators,” *Eng Appl Artif Intel* **130**, 107693 (2024).

[58] Y. Sun, A. Abudula, H. Yang, S.-S. Chiang, Z. Wan, S. Ozel, R. Hall, E. Skorina, M. Luo and C. D. Onal, “Soft mobile robots: A review of soft robotic locomotion modes,” *Current Robot Rep* **2**(4), 371–397 (2021).

[59] B. Kim, J. Ha, F. Park and P. Dupont, “Optimizing Curvature Sensor Placement for Fast, Accurate Shape Sensing of Continuum Robots,” **In:** *2014 IEEE International Conference on Robotics and Automation (ICRA)*, (IEEE, 2014) pp. 5374–5379.

[60] K. Deb, “Multi-objective genetic algorithms: Problem difficulties and construction of test problems,” *Evol Comput* **7**(3), 205–230 (1999).

[61] K. Miettinen, *Nonlinear Multiobjective Optimization*, vol. 12 (Springer Science & Business Media, Berlin, Germany, 1999).

[62] D. Kalyanmoy, “Evolutionary algorithms for multi-criterion optimization in engineering design,” *Evol Algorith Eng Comp Sci* **2**, 135–161 (1999).

[63] M. Hartikainen, K. Miettinen and M. Wiecek, “Paint: Pareto front interpolation for nonlinear multiobjective optimization,” *Comput Optim Appl* **52**(3), 845–867 (2012).

[64] K. Deb, *Optimization for Engineering Design: Algorithms and Examples* (PHI Learning Pvt. Ltd, Delhi, 2012).

[65] H. Norde, F. Patrone and S. Tijs, “Characterizing properties of approximate solutions for optimization problems,” *Math Soc Sci* **40**(3), 297–311 (2000).

[66] J.-F. Bonnans, J. C. Gilbert, C. Lemaréchal and C. A. Sagastizábal, *Numerical Optimization: Theoretical and Practical Aspects* (Springer Science & Business Media, Berlin, Germany, 2006).

[67] Y. Nomaguchi, K. Kawakami, K. Fujita, Y. Kishita, K. Hara and M. Uwasu, “Robust design of systems using uncertainty assessment based on lattice point approach: Case study of distributed generation system design in a Japanese dormitory town,” *Int J Autom Technol* **10**(5), 678–689 (2016).

[68] B. Dizangian and M. R. Ghasemi, “Reliability-based design optimization of complex functions using self-adaptive particle swarm optimization method,” *Int J Optim Civil Eng* **5**(2), 151–165 (2015).

[69] S. Edelkamp and S. Schrödl. *Heuristic Search: Theory and Applications* (Elsevier, Amsterdam, Netherlands, 2011).

[70] P. E. Hart, N. J. Nilsson and B. Raphael, “A formal basis for the heuristic determination of minimum cost paths,” *IEEE Trans Syst Sci Cybern* **4**(2), 100–107 (1968).

[71] S. J. Russell and P. Norvig, *Artificial Intelligence a Modern Approach* (Pearson, London, 2010).

[72] D. Dumitrescu, B. Lazzerini, L. C. Jain and A. Dumitrescu, *Evolutionary Computation* (CRC press, Boca Raton, 2000).

[73] D. B. Fogel, “What is evolutionary computation?,” *IEEE Spect* **37**(2), 26–32 (2000).

[74] G. Wu, R. Mallipeddi and P. N. Suganthan, “Ensemble strategies for population-based optimization algorithms–a survey,” *Swarm Evol Comput* **44**, 695–711 (2019).

[75] K. Deb, *Multi-Objective Optimisation Using Evolutionary Algorithms: An Introduction* (Springer, London, UK, 2011).

[76] H. T. Y. Baris Akbas, A. Soylemez, M. E. Zyada, M. Sarac and F. Stroppa, “The Impact of Evolutionary Computation on Robotic Design: A Case Study with an Underactuated Hand Exoskeleton,” **In:** *2024 International Conference on Robotics and Automation (ICRA)*, (IEEE, 2024).

[77] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning* (Addison-Wesley, Boston, MA, 1989).

[78] G. David, *Real-Coded Genetic Algorithms, Virtual Alphabets and Blocking* (Citeseer, Princeton, NJ, 1990).

[79] K. Deb, A. Pratap, S. Agarwal and T. A. M. T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: Nsga-ii,” *IEEE Trans Evolut Comput* **6**(2), 182–197 (2002).

[80] K. Deb and H. Jain, “An evolutionary many-objective optimization algorithm using reference-point-based non-dominated sorting approach, part i: Solving problems with box constraints,” *IEEE Trans Evolut Comput* **18**(4), 577–601 (2013).

[81] E. Zitzler, M. Laumanns and L. Thiele, “Spea2: Improving the strength Pareto evolutionary algorithm,” *TIK-Rep* **103** (2001).

[82] J. Kennedy and R. Eberhart, “Particle Swarm Optimization,” **In:** *Proceedings of ICNN’95-international conference on neural networks*, (IEEE, 1995) pp. 1942–1948.

[83] R. Storn and K. Price, “Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces,” *J Global Optim* **11**(4), 341–359 (1997).

[84] P. Larrañaga and J. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, vol. 2 (Springer Science & Business Media, Berlin, Germany, 2012).

[85] N. Hansen, “The Cma Evolution Strategy: A Comparing Review,” **In:** *Towards a New Evolutionary Computation: Advances in the Estimation of Distribution Algorithms*, (2006) pp. 75–102.
[86] N. Hansen and A. Ostermeier, “Completely derandomized self-adaptation in evolution strategies,” *Evol Comput* **9**(2), 159–195 (2001).
[87] K. Stanley and R. Miikkulainen, “Evolving neural networks through augmenting topologies,” *Evol Comput* **10**(2), 99–127 (2002).
[88] K. O. Stanley, “Compositional pattern producing networks: A novel abstraction of development,” *Genet Program Evol Mach* **8**(2), 131–162 (2007).
[89] W. H. van Willigen, E. Haasdijk and L. J. H. M. Kester, “Evolving Intelligent Vehicle Control using Multi-Objective Neat,” **In:** *2013 IEEE Symposium on Computational Intelligence in Vehicles and Transportation Systems (CIVTS)*, (IEEE, 2013) pp. 9–15.
[90] E. Medvet, A. Bartoli, F. Pigozzi and M. Rochelli, “Biodiversity in Evolved Voxel-Based Soft Robots,” **In:** *Proceedings of the Genetic and Evolutionary Computation Conference*, (2021) pp. 129–137.
[91] J. Lehman and K. O. Stanley, “Exploiting Open-Endedness to Solve Problems through the Search for Novelty,” **In:** *ALIFE*, (2008) pp. 329–336.
[92] M. Schmidt and H. Lipson, “Age-Fitness Pareto Optimization,” **In:** *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, (2010) pp. 543–544.
[93] E. Zhou and J. Hu, “Gradient-based adaptive stochastic search for non-differentiable optimization,” *IEEE Trans Automat Contr* **59**(7), 1818–1832 (2014).
[94] R. J. Balling and S. A. Wilson, “The Maximin Fitness Function for Multi-Objective Evolutionary Computation: Application to City Planning,” **In:** *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, (2001) pp. 1079–1084.
[95] X. Sui, T. Zheng, J. Qi, Z. Yang, N. Zhao, J. Zhao, H. Cai and Y. Zhu, “Task-oriented hierarchical control of modular soft robots with external vision guidance,” *J Bionic Eng* **19**(3), 657–667 (2022).
[96] B. Berger, A. Andino, A. Danise and J. Rieffel, “Growing and Evolving Vibrationally Actuated Soft Robots,” **In:** *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, (2015) pp. 1221–1224.
[97] D. Marzougui, M. Biondina and F. Wyffels, “A Comparative Analysis on Genome Pleiotropy for Evolved Soft Robots,” **In:** *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, (2022) pp. 136–139.
[98] J. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence* (MIT Press, Boston, MA, 1992).
[99] M. Killpack, D. Bodily and T. Allen, “Multi-Objective Design Optimization of a Soft, Pneumatic Robot,” **In:** *IEEE International Conference on Robotics and Automation (ICRA)*, (IEEE, 2017).
[100] L. Fathurrohim, L. R. Zuhal, P. S. Palar and Y. B. Dwianto, “Maximizing the thrust performance of flexible caudal fin panels via experimental optimization,” *Ocean Eng* **266**, 112969 (2022).
[101] C. E. Rasmussen, “Gaussian Processes in Machine Learning,” **In:** *Summer School On Machine Learning*, (Springer, 2003) pp. 63–71.
[102] B. Selman and C. P. Gomes, “Hill-climbing search,” *Encyclop Cogn Sci* **81**, 82 (2006).
[103] J. Liu, J. H. Low, Q. Han, M. Lim, D. Lu, C.-H. Yeow and Z. Liu, “Simulation data driven design optimization for reconfigurable soft gripper system,” *IEEE Robot Autom Lett* **7**(2), 5803–5810 (2022).
[104] S. Kriegman, N. Cheney, F. Corucci and J. Bongard, “A Minimal Developmental Model can Increase Evolvability in Soft Robots,” **In:** *Proceedings of the Genetic and Evolutionary Computation Conference*, (2017) pp. 131–138.
[105] J. Hiller and H. Lipson, “Automatic design and manufacture of soft robots,” *IEEE Trans Robot* **28**(2), 457–466 (2011).
[106] J. Rieffel, F. Saunders, S. Nadimpalli, H. Zhou, S. Hassoun, J. Rife and B. Trimmer, “Evolving Soft Robotic Locomotion in physx,” **In:** *Proceedings of the 11th annual conference companion on genetic and evolutionary computation conference: Late breaking papers*, (2009) pp. 2499–2504.
[107] V. P. Dinakaran, M. P. Balasubramaniyan, S. Muthusamy and H. Panchal, “Performa of scara based intelligent 3 axis robotic soft gripper for enhanced material handling,” *Adv Eng Softw* **176**, 103366 (2023).
[108] M. Chikhaoui, J. Granna, J. Starke and J. Burgner-Kahrs, “Toward motion coordination control and design optimization for dual-arm concentric tube continuum robots,” *IEEE Robot Autom Lett* **3**(3), 1793–1800 (2018).
[109] S. Djeffal, A. Amouri and C. Mahfoudi, “Kinematics modeling and simulation analysis of variable curvature kinematics continuum robots,” *UPB Sci Bull Series D Mech Eng* **83**, 28–42 (2021).
[110] A. Merrad, A. Amouri, A. Cherfia and S. Djeffal, “A reliable algorithm for obtaining all-inclusive inverse kinematics’ solutions and redundancy resolution of continuum robots,” *Arab J Sci Eng* **48**(3), 3351–3366 (2023).
[111] L. Chen, Y. Ma, Y. Zhang and J. Liu, “Obstacle avoidance and multitarget tracking of a super redundant modular manipulator based on bezier curve and particle swarm optimization,” *Chin J Mech Eng* **33**(1), 1–19 (2020).
[112] X. Gao, L. Hao, H. Yang, H. Cheng and C. Xiang, “Kinematics Solution of Hybrid Manipulator Based on pso Algorithm,” **In:** *2017 IEEE 7th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, (IEEE, 2017) pp. 199–204.
[113] H. Cheong, M. Ebrahimi and T. Duggan, “Optimal design of continuum robots with reachability constraints,” *IEEE Robot Autom Lett* **6**(2), 3902–3909 (2021).
[114] N. Tan, X. Gu and H. Ren, “Simultaneous robot-world, sensor-tip, and kinematics calibration of an underactuated robotic hand with soft fingers,” *IEEE Access* **6**, 22705–22715 (2017).

[115] E. Rodriguez, B. N. Saha, J. Romero-Hdz and D. Ortega, "A multiobjective differential evolution algorithm for robot inverse kinematics," *SSRG Int J Comp Sci Eng* **3**(5), 71–79 (2016).

[116] A. Ferigo, G. Iacca and E. Medvet, "Beyond Body Shape and Brain: Evolving the Sensory Apparatus of Voxel-Based Soft Robots," **In:** *International Conference on the Applications of Evolutionary Computation (Part of EvoStar)*, (Springer, 2021) pp. 210–226.

[117] G. Methenitis, D. Hennes, D. Izzo and A. Visser, "Novelty Search for Soft Robotic Space Exploration," **In:** *Proceedings of the 2015 annual conference on Genetic and Evolutionary Computation*, (2015) pp. 193–200.

[118] I. Exarchos, K. Wang, B. Do, F. Stroppa, M. Coad, A. Okamura and C. Liu, "Task-Specific Design Optimization and Fabrication for Inflated-Beam Soft Robots with Growable Discrete Joints," **In:** *2022 International Conference on Robotics and Automation (ICRA)*, (IEEE, 2022) pp. 7145–7151.

[119] C. A. C. Coello, *Evolutionary Algorithms for Solving Multi-Objective Problems* (Springer, Berlin, Germany, 2007).

[120] R. Conor, *Reducing Premature Convergence in Evolutionary Algorithms* (University College, Ireland, Cork, 1996).

[121] N. Kubota and T. Fukuda, "Genetic algorithms with age structure," *Soft Comput* **1**(4), 155–161 (1997).

[122] X.-S. Yang, "Swarm intelligence based algorithms: A critical analysis," *Evol Intell* **7**(1), 17–28 (2014).

[123] A. Kumar and R. Gupta, "Compare the results of tuning of pid controller by using pso and ga technique for avr system," *Int J Adv Res Comp Eng Technol* **6**(2), 2130–2138 (2013).

[124] C. Ou and W. Lin, "Comparison Between PSO and GA for Parameters Optimization of PID Controller," **In:** *2006 International conference on mechatronics and automation*, (IEEE, 2006) pp. 2471–2475.

[125] M. Pelikan, D. Goldberg and E. Cantú-Paz, "Boa: The Bayesian Optimization Algorithm," **In:** *Proceedings of the genetic and evolutionary computation conference GECCO-99*, (Citeseer, 1999).

[126] I. Rechenberg, "Evolution Strategy: Nature's Way of Optimization," **In:** *Optimization: Methods and Applications, Possibilities and Limitations: Proceedings of an International Seminar Organized by Deutsche Forschungsanstalt für Luft-und Raumfahrt (DLR)*, (1989) pp. 106–126.

[127] N. Hansen, D. V. Arnold and A. Auger, "Evolution Strategies," **In:** *Springer Handbook of Computational Intelligence*, (2015) pp. 871–898.

[128] H. J. Lichtfuss, Evolution eines rohrkrümmers, (1965).

[129] H.-P. Schwefel, "Projekt Mhd-Staustrahlrohr: Experimentelle Optimierung Einer Zweiphasendüse. In Teil I," **In:** *Technischer Bericht*, vol. **11** (AEG Forschungs institute, 1968).

[130] H. Abdi, D. Valentin and B. Edelman, *Neural Networks*, (Sage, 1999).

[131] G. Squillero and A. Tonda, "Divergence of character and premature convergence: A survey of methodologies for promoting diversity in evolutionary optimization," *Inform Sciences* **329**, 782–799 (2016).

[132] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," *Nature* **521**(7553), 436–444 (2015).

[133] R. Byrd, M. Hribar and J. Nocedal, "An interior point algorithm for large-scale nonlinear programming," *Siam J Optimiz* **9**(4), 877–900 (1999).

[134] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarter Appl Math* **2**(2), 164–168 (1944).

[135] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *J Soc Ind Appl Math* **11**(2), 431–441 (1963).

[136] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Comp J* **7**(4), 308–313 (1965).

[137] A. Martin, A. Barrientos and J. D. Cerro, "The natural-ccd algorithm, a novel method to solve the inverse kinematics of hyper-redundant and soft robots," *Soft Robot* **5**(3), 242–257 (2018).

[138] S. A. Curtis, "The classification of greedy algorithms," *Sci Comput Program* **49**(1-3), 125–157 (2003).

[139] A. Aristidou and J. Lasenby, "Fabrik: A fast, iterative solver for the inverse kinematics problem," *Graph Models* **73**(5), 243–260 (2011).

[140] C. Audet and J. E. Dennis Jr, "Analysis of generalized pattern searches," *Siam J Optimiz* **13**(3), 889–903 (2002).

[141] M. J. D. Powell, *A Direct Search Optimization Method that Models the Objective and Constraint Functions by Linear Interpolation* (Springer, Berlin, Germany, 1994).

[142] P. Frazier, "Bayesian Optimization," **In:** *Recent Advances in Optimization and Modeling of Contemporary Problems*, (Informs, 2018) pp. 255–278.

[143] S. Kirkpatrick, C. D. Gelatt Jr. and M. P. Vecchi, "Optimization by simulated annealing," *Science* **220**(4598), 671–680 (1983).

[144] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock and M. Diehl, "Qpoases: A parametric active-set algorithm for quadratic programming," *Math Prog Comput* **6**(4), 327–363 (2014).

[145] H. B. Curry, "The method of steepest descent for non-linear minimization problems," *Quarter Appl Math* **2**(3), 258–261 (1944).

[146] K. Svanberg, "The method of moving asymptotes-a new method for structural optimization," *Int J Numer Meth Eng* **24**(2), 359–373 (1987).

[147] C. T. Kelley, *Solving Nonlinear Equations with Newton's Method* (Springer, Berlin, Germany, 2003).

[148] J. Nocedal and S. Wright, *Numerical Optimization* (Springer, Berlin, Germany, 1999).

[149] P. Lloyd, G. Pittiglio, J. Chandler and P. Valdastri, "Optimal Design of Soft Continuum Magnetic Robots Under Follow-the-Leader Shape Forming Actuation," **In:** *2020 International Symposium on Medical Robotics (ISMR)*, (IEEE, 2020) pp. 111–117.

[150] N. S. Usevitch, Z. M. Hammond, M. Schwager, A. M. Okamura, E. W. Hawkes and S. Follmer, "An untethered isoperimetric soft robot," *Sci Robot* **5**(40), eaaz0492 (2020).

[151] M. Wright, “The interior-point revolution in optimization: History, recent developments, and lasting consequences,” *Bull Am Math Soc* **42**(1), 39–56 (2005).

[152] J. Lai, B. Lu, Q. Zhao and H. K. Chu, “Constrained motion planning of a cable-driven soft robot with compressible curvature modeling,” *IEEE Robot Autom Lett* **7**(2), 4813–4820 (2022).

[153] J. Burgner, H. Gilbert and R. Webster, “On the Computational Design of Concentric Tube Robots: Incorporating Volume-Based Objectives,” **In:** *2013 IEEE International Conference on Robotics and Automation*, (IEEE, 2013) pp. 1193–1198.

[154] C. Bergeles, A. H. Gosline, N. V. Vasilyev, P. J. Codd, P. J. del Nido and P. E. Dupont, “Concentric tube robot design and optimization based on task and anatomical constraints,” *IEEE Trans Robot* **31**(1), 67–84 (2015).

[155] D. Rucker, B. Jones and R. Webster, “A Model for Concentric Tube Continuum Robots Under Applied Wrenches,” **In:** *2010 IEEE International Conference on Robotics and Automation*, (IEEE, 2010) pp. 1047–1052.

[156] D. C. Rucker, B. A. Jones and R. J. Webster III, “A geometrically exact model for externally loaded concentric-tube continuum robots,” *IEEE Trans Robot* **26**(5), 769–780 (2010).

[157] D. C. Rucker and R. J. Webster III, “Statics and dynamics of continuum robots with general tendon routing and external loading,” *IEEE Trans Robot* **27**(6), 1033–1044 (2011).

[158] Z. Zhang, S. Wang, D. Meng, X. Wang and B. Liang, “Soft-ccd Algorithm for Inverse Kinematics of Soft Continuum Manipulators,” **In:** *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (IEEE, 2021) pp. 639–644.

[159] H. R. Wu, J. J. Yu, J. Pan and X. Pei, “A novel obstacle avoidance heuristic algorithm of continuum robot based on fabrik,” *Sci China Technol Sci* **65**(12), 2952–2966 (2022).

[160] H. Wu, J. Yu, J. Pan, G. Li and X. Pei, “Crrik: A fast heuristic algorithm for the inverse kinematics of continuum robot,” *J Intell Robot Syst* **105**(3), 1–21 (2022).

[161] C. Bedell, J. Lock, A. Gosline and P. Dupont, “Design Optimization of Concentric Tube Robots Based on Task and Anatomical Constraints,” **In:** *2011 IEEE International Conference on Robotics and Automation*, (IEEE, 2011) pp. 398–403.

[162] T. Anor, J. Madsen and P. Dupont, “Algorithms for Design of Continuum Robots using the Concentric Tubes Approach: A Neurosurgical Example,” **In:** *2011 IEEE International Conference on Robotics and Automation*, (IEEE, 2011) pp. 667–673.

[163] Y. Adagolodjo, F. Renda and C. Duriez, “Coupling numerical deformable models in global and reduced coordinates for the simulation of the direct and the inverse kinematics of soft robots,” *IEEE Robot Autom Lett* **6**(2), 3910–3917 (2021).

[164] E. Coevoet, A. Escande and C. Duriez, “Optimization-based inverse model of soft robots with contact handling,” *IEEE Robot Autom Lett* **2**(3), 1413–1419 (2017).

[165] E. Coevoet, Y. Adagolodjo, M. Lin, C. Duriez and F. Ficuciello, “Planning of soft-rigid hybrid arms in contact with compliant environment: Application to the transrectal biopsy of the prostate,” *IEEE Robot Autom Lett* **7**(2), 4853–4860 (2022).

[166] S. F. Ghoreishi, R. D. Sochol, D. Gandhi, A. Krieger and M. Fuge, “Bayesian optimization for design of multi-actuator soft catheter robots,” *IEEE Trans Med Robot Bion* **3**(3), 725–737 (2021).

[167] S. T. Chen, Y. S. Wang, D. C. Li, F. F. Chen and X. Y. Zhu, “Enhancing interaction performance of soft pneumatic-networks grippers by skeleton topology optimization,” *Sci China Technol Sci* **64**(12), 2709–2717 (2021).

[168] S. Chen, F. Chen, Z. Cao, Y. Wang, Y. Miao, G. Gu and X. Zhu, “Topology optimization of skeleton-reinforced soft pneumatic actuators for desired motions,” *IEEE/ASME Trans Mechatr* **26**(4), 1745–1753 (2021).

[169] R. Wang, X. Zhang, B. Zhu, H. Zhang, B. Chen and H. Wang, “Topology optimization of a cable-driven soft robotic gripper,” *Struct Multidiscip Optim* **62**(5), 2749–2763 (2020).

[170] H. Morgan, J. Osorio and A. Arrieta, “Towards Open Loop Control of Soft Multistable Grippers from Energy-Based Modeling,” **In:** *2023 IEEE International Conference on Soft Robotics (RoboSoft)*, (IEEE, 2023) pp. 1–6.

[171] G. Maloisel, E. Knoop, C. Schumacher, B. Thomaszewski, M. Bächer and S. Coros, “Optimal design of flexible-link mechanisms with desired load-displacement profiles,” *IEEE Robot Autom Lett* **8**(7), 4203–4210 (2023).

[172] M. Chau, M. Fu, H. Qu and I. Ryzhov, “Simulation Optimization: A Tutorial Overview and Recent Developments in Gradient-Based Methods,” **In:** *Proceedings of the Winter Simulation Conference 2014*, (IEEE, 2014) pp. 21–35.

[173] P. Kischka, H.-W. Lorenz, U. Derigs, W. Domschke, P. Kleinschmidt, R. Möhring, J.-L. Goffin and J.-P. Vial, “Interior Point Methods for Nondifferentiable Optimization,” **In:** *Operations Research Proceedings 1997: Selected Papers of the Symposium on Operations Research (SOR’97) Jena*, (Springer, 1998) pp. 35–49. 1997.

[174] G. Dantzig, “Origins of the Simplex Method,” **In:** *A History of Scientific Computing*, (ACM, 1990) pp. 141–151.

[175] J. C. Lagarias, J. A. Reeds, M. H. Wright and P. E. Wright, “Convergence properties of the Nelder–Mead simplex method in low dimensions,” *SIAM J Optimiz* **9**(1), 112–147 (1998).

[176] J. Brown, J.-C. Latombe and K. Montgomery, “Real-time knot-tying simulation,” *Visual Comp* **20**(2-3), 165–179 (2004).

[177] M. J. Best, *An Algorithm for the Solution of the Parametric Quadratic Programming Problem* (Springer, Berlin, Germany, 1996).

[178] B. Gray, *Homotopy Theory: An Introduction to Algebraic Topology* (Academic Press, Cambridge, MA, 1975).

[179] R. T. Rockafellar, “Lagrange multipliers and optimality,” *SIAM Rev* **35**(2), 183–238 (1993).

[180] D. Camarillo, C. Milne, C. Carlson, M. Zinn and K. Salisbury, “Mechanics modeling of tendon-driven continuum manipulators,” *IEEE Trans Robot* **24**(6), 1262–1273 (2008).

[181] M. D. Gilbertson, G. McDonald, G. Korinek, J. D. Van de Ven and T. M. Kowalewski, “Serially actuated locomotion for soft robots in tube-like environments,” *IEEE Robot Autom Lett* **2**(2), 1140–1147 (2017).

[182] G. Fang, C.-D. Matte, R. Scharff, T.-H. Kwok and C. Wang, “Kinematics of soft robots by geometric computing,” *IEEE Trans Robot* **36**(4), 1272–1286 (2020).

[183] M. J. Jakiela, C. Chapman, J. Duda, A. Adewuya and K. Saitou, “Continuum structural topology design with genetic algorithms,” *Comput Method Appl Mech Eng* **186**(2-4), 339–356 (2000).

[184] C. Kane and M. Schoenauer, “Topological optimum design using genetic algorithms,” *Control Cybern* **25**(5), 1059–1088 (1996).

[185] N. Cheney, R. MacCurdy, J. Clune and H. Lipson, “Unshackling evolution: Evolving soft robots with multiple materials and a powerful generative encoding,” *ACM SIGEVOlution* **7**(1), 11–23 (2014).

[186] S. Droste, T. Jansen and I. Wegener, “On the analysis of the (1+1) evolutionary algorithm,” *Theor Comp Sci* **276**(1-2), 51–81 (2002).

[187] L. B. Booker, D. E. Goldberg and J. H. Holland, “Classifier systems and genetic algorithms,” *Artif Intell* **40**(1-3), 235–282 (1989).

[188] M. Pelikan and H. Mühlenbein, “The Bivariate Marginal Distribution Algorithm,” **In:** *Advances in Soft Computing: Engineering Design and Manufacturing*, (Springer, 1999) pp. 521–535.

[189] S. Shakya and J. McCall, “Optimization by estimation of distribution with DEUM framework based on Markov random fields,” *Int J Autom Comput* **4**(3), 262–272 (2007).

[190] S. Shakya, R. Santana and J. A. Lozano, “A markovianity based optimisation algorithm,” *Genet Program Evol Mach* **13**(2), 159–195 (2012).

[191] M. S. R. Martins, M. E. Yafrani, R. Santana, M. Delgado, R. Lüders and B. Ahiod, “On the Performance of Multi-Objective Estimation of Distribution Algorithms for Combinatorial Problems,” **In:** *2018 IEEE Congress On Evolutionary Computation (CEC)*, (IEEE, 2018) pp. 1–8.

[192] H. Cheong, M. Ebrahimi, A. Butscher and F. Iorio, “Configuration Design of Mechanical Assemblies Using an Estimation of Distribution Algorithm and Constraint Programming,” **In:** *2019 IEEE Congress On Evolutionary Computation (CEC)*, (IEEE, 2019) pp. 2339–2346.

[193] M. Dorigo, M. Birattari and T. Stutzle, “Ant colony optimization,” *IEEE Comput Intell Mag* **1**(4), 28–39 (2006).

[194] D. Izzo, M. Ruciński and F. Biscani, “The Generalized Island Model,” **In:** *Parallel Architectures and Bioinspired Algorithms*, (Springer, 2012) pp. 151–169.

[195] E. Alba and B.é Dorronsoro, “Introduction to Cellular Genetic Algorithms,” **In:** *Cellular Genetic Algorithms*, (Springer, 2008) pp. 3–20.

[196] A. Kashtiban, S. Khanmohammadi and K. Asghari, “Solving multimodal optimization problems based on efficient partitioning of genotypic search space,” *Turk J Electr Eng Comp Sci* **24**(2), 621–638 (2016).

[197] R.é Thomsen, P. Rickers and T. Krink, “A Religion-Based Spatial Model for Evolutionary Algorithms,” **In:** *Parallel Problem Solving from Nature PPSN VI: 6th International Conference*, (Springer, 2000) pp. 817–826.

[198] Y. Liang and K.-S. Leung, “Genetic algorithm with adaptive elitist-population strategies for multimodal function optimization,” *Appl Soft Comput* **11**(2), 2017–2034 (2011).

[199] D. E. Goldberg and J. Richardson, “Genetic Algorithms with Sharing for Multimodal Function Optimization,” **In:** *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, (1987) pp. 41–49.

[200] B. L. Miller and M. J. Shaw, “Genetic Algorithms with Dynamic Niche Sharing for Multimodal Function Optimization,” **In:** *International Conference on Evolutionary Computation*, (IEEE, 1996) pp. 786–791.

[201] A. Pétrowski, “A Clearing Procedure as a Niching Method for Genetic Algorithms,” **In:** *International Conference on Evolutionary Computation*, (IEEE, 1996) pp. 798–803.

[202] R. Thomsen, “Multimodal Optimization Using Crowding-Based Differential Evolution,” **In:** *Congress on Evolutionary Computation*, (IEEE, 2004) pp. 1382–1389.

[203] X. Yin and N. Germay, “A Fast Genetic Algorithm with Sharing Scheme using Cluster Analysis Methods in Multimodal Function Optimization,” **In:** *Artificial Neural Nets and Genetic Algorithms: International Conference*, (Springer, 1993) pp. 450–457.

[204] K. Deb and A. Saha, “Multimodal optimization using a bi-objective evolutionary algorithm,” *Evol Comput* **20**(1), 27–62 (2012).

[205] K. E. Yenin, R. O. Sayin, K. Arar, K. K. Atalay and F. Stroppa, “Multi-modal optimization with k-cluster big bang-big crunch algorithm” (2023). arXiv preprint arXiv: 2401.06153.

[206] R. T. F. A. King, K. Deb and H. C. S. Rughooputh, “Comparison of nsga-ii and spea2 on the multiobjective environmental/economic dispatch problem,” *Univ Maurit Res J* **16**(1), 485–511 (2010).

[207] H. Calborean, R. Jahr, T. Ungerer and L. Vintan, “A Comparison of Multi-Objective Algorithms for the Automatic Design Space Exploration of a Superscalar System,” **In:** *Advances in Intelligent Control Systems and Computer Science* (Springer, 2013) pp. 489–502.

Cite this article: F. Stroppa, F. J. Majeed, J. Batiya, E. Baran and M. Sarac, “Optimizing soft robot design and tracking with and without evolutionary computation: an intensive survey”, Robotica. https://doi.org/10.1017/S0263574724001152