2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 1-5, 2023. Detroit, USA

Computational Design of Closed-Chain Linkages: Respawn Algorithm for Generative Design

Dmitriy V. Ivolga[1], Ivan I. Borisov[1], Kirill V. Nasonov[1], Sergey A. Kolyubin[1]

***Abstract*— Designing robots is a multiphase process aimed at solving a multi-criteria optimization problem to find the best possible detailed design. Generative design (GD) aims to accelerate the design process compared to manual design, since GD allows exploring and exploiting the vast design space more efficiently. In the field of robotics, however, relevant research focuses mostly on the generation of fully-actuated open chain kinematics, which is trivial in mechanical engineering perspective. Within this paper, we address the problem of generative design of closed-chain linkage mechanisms. A GD algorithm has to be able to generate meaningful mechanisms which satisfy conditions of existence. We propose an optimization-driven algorithm for generation of planar closed-chain linkages to follow a predefined trajectory. The algorithm creates an unlimited range of physically reproducible design alternatives that can be further tested in simulation. These tests could be done in order to find solutions that satisfy extra criteria, e.g., desired dynamic behavior or low energy consumption. The proposed algorithm is called "respawn" since it builds a new linkage after the ancestor has been tested in a virtual environment in pursuit for the optimal solution. To show that the algorithm is general enough, we show a set of generated linkages that can be used for a wide class of robots.**

I. Introduction

Designing of robots is a multistep process through which a set of high-level task objectives are first mapped onto a morphology, then to a mechanical structure, and finally to a detailed design. There are multiple approaches to how to do design, such as *intuitive* or *bioinspired manual design*, *model-based design optimization*, *generative* and *evolutionary* design. Manual design approach suffers from significant input from a designer, thus we can not say much about optimally of the results [1]. Biomimetics can give useful insights, but also restrict the space of solutions. Model-based design optimization, as well as evolutionary and generative design (GD) approaches are capable more efficiently to explore and exploit the vast design space, but suffer from the curse of dimensionality of the design space [2].

GD is intended to produce mechanically improved solutions based on robust and rigorous models of design conditions and performance criteria [3]. GD has to shorten the time for synthesis and verification to find better solutions. The results should surpass the design solutions provided by experts, or at least be on par with them.

*This work was supported by the Analytical Center for the Government of the Russian Federation (IGK 000000D730321P5Q0002), agreement No. 70-2021-00141.

[1]Dmitriy V. Ivolga, Kirill V. Nasonov, Ivan I. Borisov, and Sergey A. Kolyubin are with the Biomechatronics and Energy-Efficient Robotics Lab, ITMO University, Saint Petersburg, Russia e-mail: {ivolga.dv, kvnasonov, borisovii, s.kolyubin}@itmo.ru

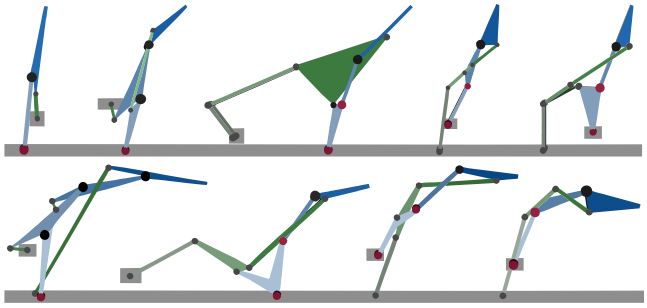

Fig. 1. A set of generated linkages with closed kinematics. Red circles indicate actuated joints, while black and grey circles represent constrained joints. The black circles connect the initial open chain marked in blue, while the grey circles connect the attached group of links marked in green

A lot of efforts have been spent on utilizing finite element analysis to generate rigid and elastic bodies [4]. Both rigid and elastic bodies passively experience external loads and could be considered as static and quasi-static cases respectively; generative design of such systems is a mature technology and even commercially available [5]. On the other hand, the problem of generative design of dynamic systems that transform motion of joints into complex links motion is more challenging and is still far away from being solved.

A. Related work

Model-based design optimization approaches can be used as a tool by a user to efficiently synthesize linkage together with needed control. Gradient-based together with global optimization algorithms are used to optimize geometry, mass distribution, or actuation within specific boundaries [6], [7]. Properly formalized optimization tasks allow to efficiently find suitable solutions, however model-based design optimization approaches strongly depend on user inputs.

As an alternative approach, the evolutionary design needs minimal input from the user to create unique and unexpected solutions. However, the lack of the current simulation software the evolutionary design creates highly abstract structures that hardly can be physically reproduced [8], [9], [10].

GD is intended to automatize co-optimization of mechanical structure and joints trajectories to satisfy poorly formalized tasks from a user. Generative adversarial networks (GANs) [11], graph neural networks (GNNs) [12], [13], and deep neural networks (DNNs) [14] are used to generate robots for a wide range of scenarios. While those systems can generate fancy creatures that could be physically reproduced, the question remains whether such systems are efficient in some sense compared to the morphologies found manually.

2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-9190-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/IROS55552.2023.10341425

Authorized licensed use limited to: St Petersburg Natl Uni of Info Tech Mech & Optics. Downloaded on October 07,2024 at 10:20:49 UTC from IEEE Xplore. Restrictions apply.

B. Morphological computation

The generated mechanisms have to be physically reproducible, and their implementation should be justified by superior characteristics compared to previously manually synthesized mechanisms. We believe that the *morphological computation* or *mechanical intelligence* is the way to ensure the characteristics' superiority by means of proper mechanics that shape the desired behavior.

Morphological computation of control laws is a principle, according to which the major portion of robots' desired behavior can be achieved with the "body" instead of the "brain". Morphological computation allows designing efficient robots, but the design process itself is non-formalized. Although there are guidelines for design, the overall process is by its very nature as much as art as science. The process of designing such structures is often informal due to the combination of components of different physical nature in a robotic system, many topologies and parameters tending to infinity, and heterogeneous criteria for optimal solutions, nonlinear objective functions, etc. The result of a design process strongly depends on the developer's creativity, imagination, engineering intuition, judgment, and experience.

C. Contribution & overview

In [15] we have presented a synthesis method for reconfigurable closed-chain underactuated linkages for robotic systems that physically interact with the environment. The emphasis was made on the synthesis of a mechanism *structure* and application of underactuated adaptive grasping. The method consists of three stages: (1) *co-design of a fully-actuated open chain mechanism*, (2) the following *mechanism closure* to get a closed chain linkage that can follow the original motion, and (3) subsequent *allocation of actuators and elastic elements* in joints to get an efficient mechanism.

Different prototypes have been created to study the benefits and practical applications of the method in engineering practice: an adaptive artificial hand [16], an energy-efficient hopping robot [17], and an ergonomic exosuit [18]. However, those prototypes can not be named as optimal, since the vast design space has not been properly explored.

Within this paper, we want to make a step toward extending and clarifying the second stage of the synthesis method in the scope of generative design. While the generation of open chain mechanics is just a sequential links' connection in a series, closed chains impose requests on mechanism existence. Besides the structure problem on what kind of links groups to be added to an initial open chain, here we specify the parametric optimization of attached links geometry and calculation where it should be attached.

The method is inspired by the method presented in [6], but instead of recursively add one link at a time, we propose adding a group of links. That approach allows to generate multiple closed chain linkages with multiple closed loops that fit the idea of generative design. The idea is to able to generate unlimited number of mechanisms that can be further tested in simulation environment according to the inverse design problem.

Closed-chain kinematics allows to shape mechanism behavior by means of holonomic constraints. The generated results have to have the following advantages over generic open kinematics (1) it needs less number of drives and/or lower performance drives; (2) because of morphological computation, the algorithmic control part is simpler, and (3) minimization of dimensions, weight and total cost primary because of less number of drives.

While within this paper we focus on the kinematics aspect only, i.e. structure, geometric parameters, holonomic constraints, the synthesis could be done in terms of dynamic parameters such as mass distribution and elasticity, but those are beyond the scope of the paper.

We start with a short overview of the method, then proceed with a parametric optimization algorithm, and wrap the paper with simulation of generated mechanisms.

II. Design loop

A. The General Method

For the sake of clarity we give here a short version of the proposed method that consists of the following three steps.

1) Co-design of a fully actuated open-chain: mechanism that interacts with the environment. At the first stage, the goal is to find the number of links, their geometric parameters, attachment points to the body, and also to find proper joints trajectories to accomplish the desired behavior. The first step is needed to generate a reference for closed kinematics. While the task is trivial in a sense of mechanical engineering, the problem is to set tradeoffs between the exploration and exploitation of the vast design space to find a sub-optimal solution in meaningful time.

2) Closure: of the generated open chain kinematics. At the second stage, the goal is to find the proper structure and geometric parameters of the link groups to be attached. The idea is to relocate motors and/or to decrease their number by means of holonomic constraints. Here we look for structures capable of following the trajectories of initially generated open-chain. While the precise trajectory following is not always the desired behavior, here we look for a mechanism with the same workspace. It is possible to synthesize a wide range of closed chain kinematics for a specific motion. That stage is the main interest of that paper.

3) Optimization of actuators: for a set of generated closed-chain linkages. At the second stage, the goal is to find an optimal actuators' allocation and their motion efficient in terms of generated force and needed energy. By actuators, we consider both active drives and passive elastic elements. The final generated result could be either fully or underactuated.

B. Structural Synthesis

The design process begins with co-design or co-optimization of an open chain mechanism to generate a closed chain mechanism together with trajectory reference. Fig. 2 shows the basic intuition example of synthesis of a closed-chain mechanism out of an open chain with two links and two revolute joints (Fig. 2, a).

Authorized licensed use limited to: St Petersburg Natl Uni of Info Tech Mech & Optics. Downloaded on October 07,2024 at 10:20:49 UTC from IEEE Xplore. Restrictions apply.

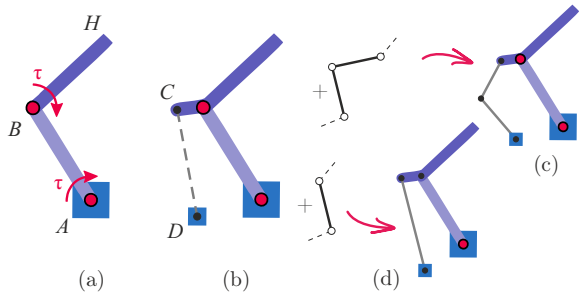

Fig. 2. An intuition of the proposed structural synthesis algorithm for a planar 2R mechanism: (a) preliminary optimized open-chain mechanism for reference, (b) to form a closed-chain linkage a group of extra links has to bound end-effector with base, (c) is a trivial way to add a group of links to form a 5 bar linkage and (d) is a trivial way to decrease DoF by adding a holonomic constrain to form a 4 bar linkage. Red circles indicate actuators

To compute degrees of freedom of a planar mechanism we need to use Chebychev–Grübler criterion

$$DoF = m \cdot n - \sum_{i=1}^{J} c_i,$$

where $m = 3$ is the number of degrees of freedom of a rigid body within a planar mechanism, n is the number of movable links, and c_i is the number of constraints provided by joint i. For a planar 2R mechanism we have 2 degrees of freedom, since $\sum_{i=1}^{J} c_i = 2P$, where P is a number of revolute joints which impose two constraints.

The Chebychev–Grübler criterion can be used both to compute DoF for a mechanism and for groups of links to be attached to the mechanism in order to impose holonomical constraints. Thus, we can compute what groups of links could be attached in order to relocate actuators and/or to decrease their number. For example, if we want to keep the number of drives intact, DoF of links group must be equal to zero

$$3 \cdot n - 2P = 0, \rightarrow P = 3 \cdot n/2,$$

where the trivial combination is a group with two links $n = 2$ and three revolute joints $P = 3$ (Fig. 2, c). In case if we want to eliminate one drive we need to eliminate one DoF

$$3 \cdot n - 2P = -1, \rightarrow P = (3 \cdot n + 1)/2,$$

where the trivial combination is a one link $n = 1$ with two revolute joints $P = 2$ (Fig. 2, d), however we can attach a group of three links $n = 3$ with five revolute joints $P = 5$ to eliminate the same one DoF.

Since we want to synthesize closed kinematics, the links group has to bind an end-effector with a mechanism base, otherwise it would not be the closed chain mechanism (Fig. 2, b). While the example with single rigid link is the one presented in [6], we propose to attach not a single rigid link at the time (Fig. 2, d), but to attach the entire group (Fig. 2, c). Moreover, just like we can recursively attach one rigid link at a time, we can recursively attach groups of links producing multiple closed-chain linkages. Respawning of such mechanisms is needed in the scope of generative design of linkages.

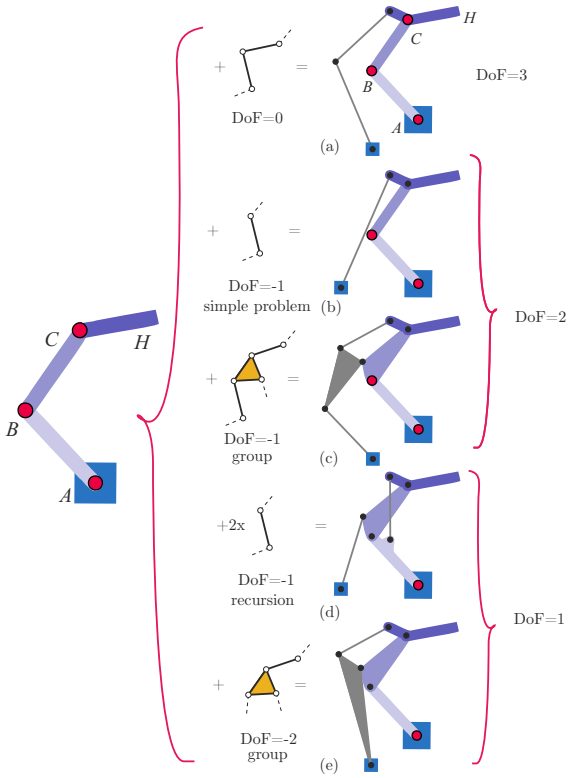

Fig. 3. An elaborated example of 3R mechanism structure synthesis

Structural synthesis of a closed chain kinematics for an original 3R robot is shown in Fig. 3. The group of two binary links $n = 2$ with three revolute joints $P = 3$ does not change the initial DoF (Fig. 3, a); the group connects the end-effector with the base. The trivial "group" of single binary link $n = 1$ with two revolute joints $P = 3$ eliminates single DoF (Fig. 3, b). The same DoF could be eliminated with the next in line group of three links $n = 3$ with five revolute joints $P = 5$ (Fig. 3, c). Here the ternary link could be both in the middle of the group and at the end.

To eliminate 2 DoF we could recursively add two single links according to [6] (Fig. 3, d), or we can add a group of two links $n = 2$ with four revolute joints $P = 4$ (Fig. 3, e). Moreover, we can recursively add two groups of three links $n = 3$ with five revolute joints $P = 5$ to eliminate the same 2 DoF, however that system could be overlinked.

C. Respawn algorithm

The algorithm proposed is based on simulation. It uses graph grammar and optimization to tackle the problem of generation of closed-chain linkages under desired conditions. Graph grammar formalizes descriptions of linkage structural components and prescribes rules by which the components could be connected to each other [12]. The overall structure of a linkage forms a directed graph that can be analyzed and converted into a simulation object. A graph is changed by adding new structures, updating existing structures, and creating connections between nodes. The changes depend on the type of groups that are being attached.

Authorized licensed use limited to: St Petersburg Natl Uni of Info Tech Mech & Optics. Downloaded on October 07,2024 at 10:20:49 UTC from IEEE Xplore. Restrictions apply.

The algorithm goal is to synthesize a set of closed-chain linkages capable of following the reference trajectory of previously synthesized open-chain. Respawning of a linkage is needed for further tests of the synthesized linkage during the third stage of the proposed method.

In order to generate a set of design-candidates, the algorithm performs two optimization procedures during N episodes. The *inner loop* searches for *geometrical parameters of a group of links* to be attached in parallel to the initial open chain, while the *outer loop* searches for how to add holonomic constraints *to reduce degree of freedom.*

Before running the proposed algorithm, an initial open-chain linkage with a reference trajectory must be provided together with a value of open chain DoF W_i and desired value for closed chain linkage DoF W_d. Also, we have to initialize *connection points vector* $\mathbf{P}$, that describes the position where group of links to be attached touches the base link and end-effector, *closure points vector* $\mathbf{C}$, that is needed to reduce DoF, and *links lengths' vector* $\mathbf{L}$.

According to Fig. 3 there are several ways to get the desired DoF: recursively by adding a trivial "group" with $DoF = -1$, or alternatively we can attach a group with $DoF = W_i - W_d$. Graph grammar rules are used to operate with groups of links to be attached. Initial open-chain linkage has to be transformed into a graph that can be added by extra grammar to form closed-chain mechanism. Rules are applied until the mechanism DoF W is not equal to the desired value W_d. Design candidates are stored in a set $\mathscr{D}$ that contains a set of found sub-optimal candidates.

A candidate d is randomly selected from $\mathscr{D}$ to perform the optimization of geometrical parameters. The internal loop optimizes connection points $\mathbf{P}$; the criterion is minimization of Q_1 and Q_2 for each outer loop. The inner loop is limited with V iterations. The outer loop optimizes $\mathbf{C}$ and $\mathbf{L}$ and is limited with M iterations.

The inner loop uses simulation to calculate cost function value. $BuildDesign()$ function checks whether a mechanism can be assembled; in case of failure, it gets the maximum cost value. Simulation cycle $Simulation()$ is run if a generated mechanism can be assembled; actual motion $\hat{J}(t)$ is recorder as simulation result to calculate criteria and cost function. $internalOptimizer()$ compares cost function values f with the best f_i^* in the inner loop, updates $\mathbf{P}$ and initiates the next iteration. Finally, the best candidate design in the inner loop is found; the results proceeds to the outer loop. $externalOptimizer()$ updates $\mathbf{C}$ and $\mathbf{L}$; $internalOptimizer()$ and $externalOptimizer()$ operate within global optimization algorithm. After each episode the best sup-optimal design is collected in the set set n^* of the best found design.

Algorithm 1 Respawn Algorithm for Generative Design

Inputs: Initial DoF W_i, desired DoF W_d, desired trajectory $\bar{J}(t)$, number of iterations N, number of candidate designs K and optimization iterations M, V
Outputs: The best solutions found n^*
Initialize: set of best designs $n^* \leftarrow \{\}$, $W \leftarrow W_i$ the current DoF, cost function value $f \leftarrow$ MAX == **const**
for $i \leftarrow 1$ to N **do**
  $\mathscr{D} \leftarrow \{\}$ ▷ Initialize possible design candidates
  for $k \leftarrow 1$ to K **do** ▷ Computation of groups structure
    $n \leftarrow$ initial design graph
    while $W \neq W_d$ **do**
      apply random allowed grammar rule
    $\mathscr{D} \leftarrow \mathscr{D} \cup n$ ▷ Add possible candidate
  ▷ Choose one to be the candidate
  Select a random design from $\mathscr{D}$ as the candidate d
  $\mathbf{C}, \mathbf{L}, \mathbf{P} \leftarrow$ initialize vectors
  ▷ Optimize vector $\mathbf{C}$ and $\mathbf{L}$
  while iteration is less than M **do**
    ▷ Optimize vector of P
    while iteration is less than V **do**
      $break \leftarrow BuildDesign(d, \mathbf{C}, \mathbf{P}, \mathbf{L})$
      if break == True **then**
        $f \leftarrow MAX$
      else
        $\hat{J}(t) \leftarrow Simulation(d, \mathbf{C}, \mathbf{P}, \mathbf{L})$
        if n_{group} == 1 and P_{group} == 2 **then**
          $Q_1 \leftarrow \lambda$
        else
          $Q_1 \leftarrow \eta$
        $Q_2 \leftarrow LSED(\bar{J}(t), \hat{J}(t)(t)$
        $f \leftarrow \omega_1 \cdot Q_1 + \omega_2 \cdot Q_2$
      if $f < f_i^*$ **then**
        $f_i^* \leftarrow f$
      update $\mathbf{P}$ with $internalOptimizer()$
    if $f_i^* < f_e^*$ **then** ▷ Choosing the best value
      $d^*, c^*, l^*, p^*, f_e^* \leftarrow d, \mathbf{C}, \mathbf{L}, \mathbf{P}, f_i^*$
    update C, L with $externalOptimizer()$
  $n^* \leftarrow n^* \cup [d^*, c^*, l^*, p^*, f_e^*]$ ▷ Save the best design

D. Criteria

The cost function is used to numerically evaluate how a design candidate follows the predetermined trajectory. The cost function is calculated as the sum of two criteria

$$f = \omega_1 \cdot Q_1 + \omega_2 \cdot Q_2,$$

where ω_1 and ω_2 are weights coefficients.

1) Criteria Q_1: depends on links of a group to be attached. We search for a pair of points p_i and p_j that belong to different links i and j which should be connected. If the group is a trivial one (a single link $n = 1$ with two joints $P = 2$), the algorithm searches for p_I and p_j such that the distance between them tends to a constant value and varies least throughout the whole cycle of motion

$$Q_1 = \lambda = \frac{1}{N} \sum_m^N (||p_i^W(t_m) - p_j^W(t_m)||^2 - \bar{l}^2)^2,$$

where $p_i^W(t) \in \mathbb{R}^3$ and $p_j^W(t) \in \mathbb{R}^3$ are position vectors of points p_i and p_j respectively, expressed in world frame Ψ_W, N is number of discrete measurements, t_m is measurement time , $\bar{l}^2$ is squared mean distance between the points p_i and p_j per cycle.

Authorized licensed use limited to: St Petersburg Natl Uni of Info Tech Mech & Optics. Downloaded on October 07,2024 at 10:20:49 UTC from IEEE Xplore. Restrictions apply.

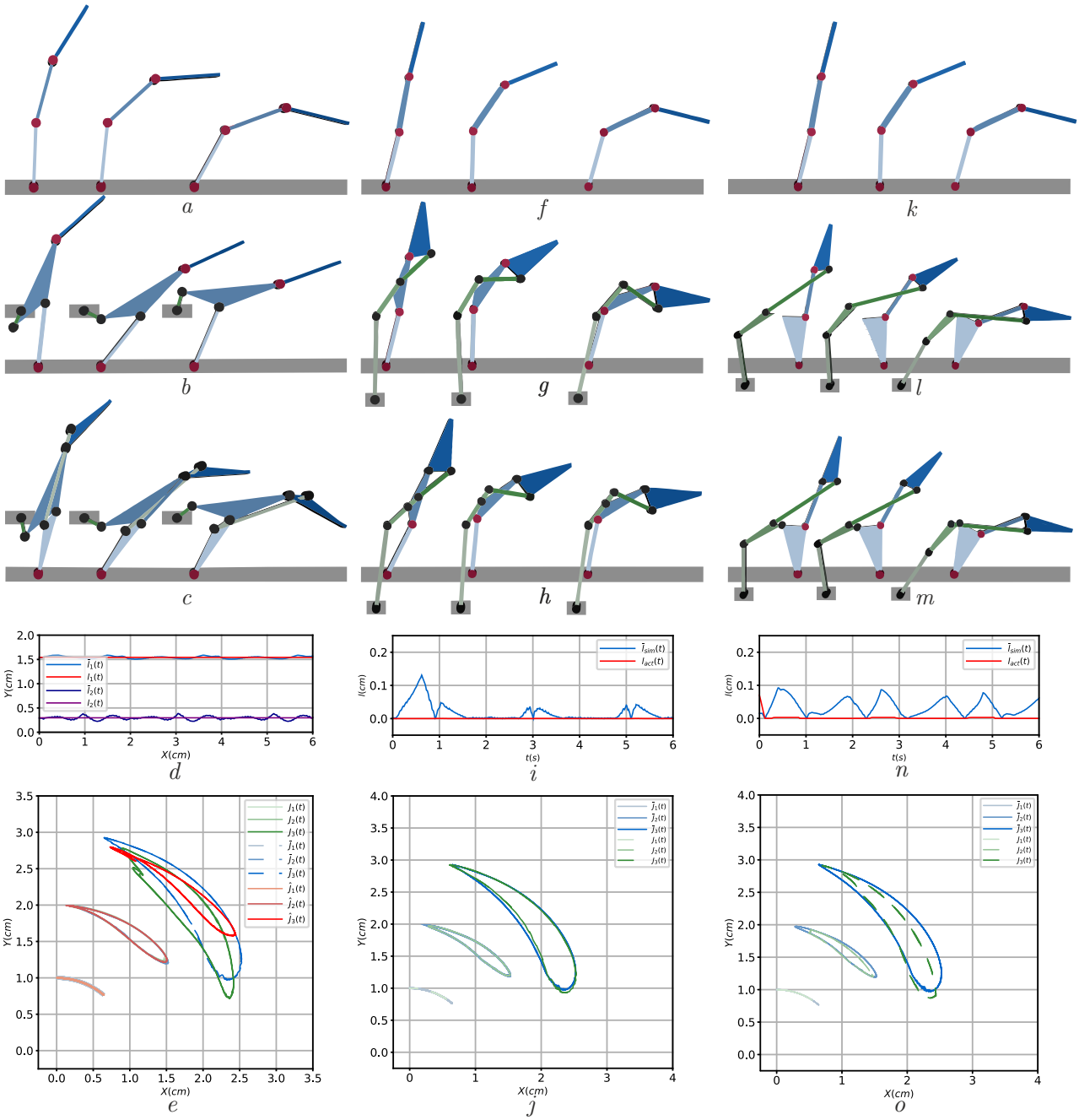

Fig. 4. Synthesis examples for a $3R$ mechanism: recursion solution by adding the trivial "group" of a single link n =1 with two revolute joints P = 2 (a - e) and variants of adding actual link groups n = 3 P = 5 (f - j and k - o). Red joints represent active motors; l is distance between the closing points, $J_1(t) - J_3(t)$ are trajectories of points on the links of the mechanism. Bar sign $\bar{J}$ indicates trajectories for initial open chain; hat sign $\hat{J}$ indicates trajectories of intermediate results of synthesis after adding first group $n = 1$ and $P = 2$, J indicates trajectories of the final closed chain linkages

If a group to be attached is actually a group, we search for a pair of points p_i and p_j such that the distance between them tends to zero

$$Q_1 = \eta = \frac{1}{N}\sum_{m}^{N} ||p_i^W(t_m) - p_j^W(t_m)||^2.$$

2) Criteria Q_2: estimates how trajectory is changed after group has been attached. Lock-step Euclidean distance LSED was used to estimate motion in Q_2 [19]. The estimation method is chosen because it doesn't require significant resources and the dimensions of the estimated trajectories are the same with a fixed sampling step

$$Q_2 = \sum_{m}^{N} \sqrt{d_{euc}(\bar{J}(t_m), \hat{J}(t_m))},$$

where $d_{euc}(q,p) = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$ is the Euclidean distance between points $q, p \in \mathbb{R}^2$; $\bar{J}(t_m), \hat{J}(t_m)$ are points of the desired and actual trajectories at time t_m, respectively.

Authorized licensed use limited to: St Petersburg Natl Uni of Info Tech Mech & Optics. Downloaded on October 07,2024 at 10:20:49 UTC from IEEE Xplore. Restrictions apply.

III. Simulation

The script was implemented in Python 3.9.12. Simulation was conducted using an open source multi-physics simulation engine PyChrono [20]. Visualization of generated linkages was implemented using Irrlicht 1.8.5. SciPy library was used to implement functions for calculating linear algebra and optimization.

The simulations of the generated kinematic structures were performed without taking into account external forces, such as contact interactions. All bodies were represented as planar, homogeneous, rigid objects. The tensor of inertia of bodies was calculated by means of the simulation environment based on a fixed density value. All active joints were actuated by means of built-in position control. To avoid having singular configurations, passive joints are bounded by angular limits.

The algorithm was run on a computer with Intel-i7 and RTX2060. The computation time of the inner optimization loop is no more than 60 seconds with a limit of 10000 iterations. Calculation of the outer optimization loop depends on the limitation of the number of iterations M.

Fig. 4 demonstrates results of the proposed algorithm for the $3R$ mechanism. Three variants are shown to form a closed-chain mechanism: recursive solution $n = 1$ $P = 2$ (Fig. 4, *a* - *c*), and two different variants with group $n = 3$ $P = 5$ (Fig. 4, *f* - *h*, *k* - *m*). Fig. 4, *d* shows that the distances between the closing points $\bar{l}$ fluctuate around constant values $l_1 = 1.54$ cm and $l_2 = 0.32$ cm, which correspond to the link lengths of the $n = 1$ $P = 2$ groups. Fig. 4, *i* and *n* show that the distances between the connected points fluctuate around 0. Fig. 4, *e*, *j*, *o* compare the repeatability of the original trajectory for all three closure methods. For each example, the values of the difference in area in percentage terms were evaluated.

IV. Conclusion

The design guidelines of robots are not formalized due to nonlinearities, holonomic connections, continuous nature of elastic deformation, unlimited spaces of structures and geometric parameters. The synthesis result strongly depends on developers' creativity, knowledge, and experience. Generative design algorithms can greatly simplify the process of synthesis of complex robotic structures. We treat GD as a tool to automatically generate mechanisms to satisfy a weakly formalized task of a software user to transform motion of actuators into complex trajectories.

In this paper, we demonstrate the optimization-driven algorithm, which allows us to automate the process of generation of closed-chain linkages. The proposed algorithm is the second step in the previously presented method for robots' co-design. The algorithm can be applied to synthesize mechanisms for a wide application. The proposed algorithm is capable of generating an unlimited range of physically reproducible designs of closed-chain linkages. In future works, we plan to extend the algorithm to generate not only planar but also spatial mechanisms.

References

[1] J. Pinskier and D. Howard, "From bioinspiration to computer generation: Developments in autonomous soft robot design," *Advanced Intelligent Systems*, vol. 4, no. 1, p. 2100086, 2022.

[2] F. Chen and M. Y. Wang, "Design optimization of soft robots: A review of the state of the art," *IEEE Robotics & Automation Magazine*, vol. 27, no. 4, pp. 27–43, 2020.

[3] H. Isakhani, N. Bellotto, Q. Fu, and S. Yue, "Generative design and fabrication of a locust-inspired gliding wing prototype for micro aerial robots," *Journal of Computational Design and Engineering*, vol. 8, no. 5, pp. 1191–1203, 2021.

[4] S. Transue and M.-H. Choi, "Generative deformation: procedural perforation for elastic structures," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3672–3678.

[5] Autodesk. Fusion 360 generative design extension. [Online]. Available: https://www.autodesk.com/products/fusion-360/generative-design-extension?term=1-YEARtab=subscription

[6] B. Thomaszewski, S. Coros, D. Gauge, V. Megaro, E. Grinspun, and M. Gross, "Computational design of linkage-based characters," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, pp. 1–9, 2014.

[7] F. De Vincenti, D. Kang, and S. Coros, "Control-aware design optimization for bio-inspired quadruped robots," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 1354–1361.

[8] N. Cheney, R. MacCurdy, J. Clune, and H. Lipson, "Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding," *ACM SIGEVOlution*, vol. 7, no. 1, pp. 11–23, 2014.

[9] F. Corucci, N. Cheney, F. Giorgio-Serchi, J. Bongard, and C. Laschi, "Evolving soft locomotion in aquatic and terrestrial environments: effects of material properties and environmental transitions," *Soft robotics*, vol. 5, no. 4, pp. 475–495, 2018.

[10] J. Bhatia, H. Jackson, Y. Tian, J. Xu, and W. Matusik, "Evolution gym: A large-scale benchmark for evolving soft robots," *Advances in Neural Information Processing Systems*, vol. 34, pp. 2201–2214, 2021.

[11] J. Hu, J. Whitman, M. Travers, and H. Choset, "Modular robot design optimization with generative adversarial networks," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 4282–4288.

[12] A. Zhao, J. Xu, M. Konaković-Luković, J. Hughes, A. Spielberg, D. Rus, and W. Matusik, "Robogrammar: graph grammar for terrain-optimized robot design," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–16, 2020.

[13] T. Wang, Y. Zhou, S. Fidler, and J. Ba, "Neural graph evolution: Towards efficient automatic robot design," *arXiv preprint arXiv:1906.05370*, 2019.

[14] A. Gupta, L. Fan, S. Ganguli, and L. Fei-Fei, "Metamorph: Learning universal controllers with transformers," *arXiv preprint arXiv:2203.11931*, 2022.

[15] I. I. Borisov, E. E. Khomutov, S. A. Kolyubin, and S. Stramigioli, "Computational design of reconfigurable underactuated linkages for adaptive grippers," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 6117–6123.

[16] I. I. Borisov, E. E. Khomutov, D. V. Ivolga, N. A. Molchanov, I. A. Maksimov, and S. A. Kolyubin, "Reconfigurable underactuated adaptive gripper designed by morphological computation," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 1130–1136.

[17] K. V. Nasonov, D. V. Ivolga, I. I. Borisov, and S. A. Kolyubin, "Computational design of closed-chain linkages: Hopping robot driven by morphological computation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 7419–7425.

[18] O. V. Borisova, I. I. Borisov, K. A. Nuzhdin, A. M. Ledykov, and S. A. Kolyubin, "Computational design of closed-chain linkages: synthesis of ergonomic spine support module of exosuit," , vol. 14, no. 6, pp. 1269–1280, 2022.

[19] Y. Tao, A. Both, R. Silveira, K. Buchin, S. Sijben, R. Purves, P. Laube, D. Peng, K. Toohey, and M. Duckham, "A comparative analysis of trajectory similarity measures," *GIScience Remote Sensing*, vol. 58, pp. 1–27, 06 2021.

[20] A. Tasora, R. Serban, H. Mazhar, A. Pazouki, D. Melanz, J. Fleischmann, M. Taylor, H. Sugiyama, and D. Negrut, "Chrono: An open source multi-physics dynamics engine," 06 2016, pp. 19–49.

Authorized licensed use limited to: St Petersburg Natl Uni of Info Tech Mech & Optics. Downloaded on October 07,2024 at 10:20:49 UTC from IEEE Xplore. Restrictions apply.