An Optimization Study on Modular Reconfigurable Robots: Finding the Task-Optimal Design

Edoardo Romiti, Francesco Iacobelli, Marco Ruzzon, Navvab Kashiri, Jörn Malzahn and Nikos Tsagarakis

2023 IEEE 19th International Conference on Automation Science and Engineering (CASE) | 979-8-3503-2069-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/CASE56687.2023.10260507

Abstract— **Reconfigurable collaborative robots are adaptive robotic systems offering new opportunities to rapidly create fit-to-task flexible automation lines demanded by the recent increasingly varying market needs in low-volume/high-mix industrial manufacturing. In this context, the configuration of a reconfigurable robot, comprising its morphology and its base placement, can be optimized for a certain criterion anytime the need for a change in batch production arises, boosting performances with respect to conventional fixed-structure robots.**

In this work we study and analyze how to find the best-suited reconfigurable robot topology for a given task, starting from a fixed set of joint and link modules. We make use of a two-stage generative design optimization approach to avoid running into issues from the "curse of dimensionality". The efficacy of the approach is investigated and validated in a sequence of peg-in-hole tasks with a minimum-effort objective function. Simulation results are verified against real-world experiments with an in-house developed reconfigurable robot prototype comparing optimal against sub-optimal robot configurations.

I. INTRODUCTION

Recent manufacturing trends request for reconfigurable manufacturing systems (RMS) to easily and rapidly scale and re-purpose manufacturing capacity in response to market evolving demands and fluctuations, frequently varying products and accelerating product life-cycles [1]. Generative design optimization tools for RMS are required to leverage the full adaptation potential of RMS [2], [3], without a cost and time intensive engineering process. This article shows how this process can be automatized by using a two-staged optimization scheme for fit-to-task generative optimization for reconfigurable robots, representing a special class of RMS. The system depicted in Fig. 1 is a laboratory prototype of such a system. Being composed of "plug & work" hardware modules, the user can instantly and reversibly disassemble and reassemble them to realize task-optimal kinematic arrangements within only minutes. Given a description of a new task, an available set of modules and a full characterization of the functionality and constraints for each module along with the cost criteria, the proposed optimization scheme generates the best combination of modules (morphology) along with their physical placement in the workspace and optimal motion sequence to execute a new task. This article combines the discrete optimal morphology and placement problem with the continuous optimal task execution problem.

Most works in the related literature are restricted to one of these problems such as motion planning [4]. Regarding

The authors are with the Humanoids and Human-centered Mechatronics Lab, Istituto Italiano di Tecnologia, Genova, Italy edoardo.romiti@iit.it

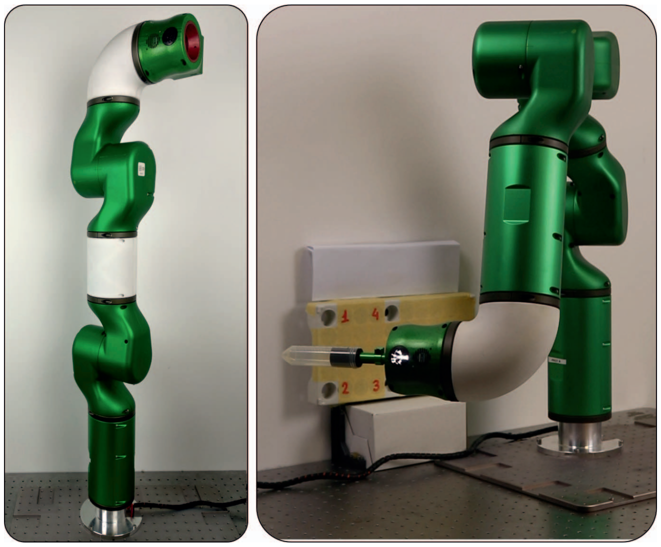


Fig. 1: Two morphology examples for the reconfigurable prototype.

the robot design, the authors of [5] introduce a task-based robot design for a fixed morphological joint-link sequence. The formulation assumes all design parameters such as link lengths and offsets to be continuous values, leading to a continuous nonlinear optimization problem for which standard solutions are available [6]. This approach is typically referred to as "kinematic synthesis" and was used also in [7], [8].

The main challenge for the discrete optimal robot morphology problem arises from the explosive growth in the number of possible configurations with the variety of modules and the number of degrees of freedom required by the task. It may become infeasible to find the optimal morphology in reasonable time. The early work in [9] proposes a "decision policy" for finding the discrete robot morphology respecting a set of "task work" points. Similarly, [10] introduces a heuristic approach to the discrete optimal robot morphology problem for a motion task provided by the user. The approach in [11] first constraints the morphological structure to conventional kinematics used in industrial robots and then follows a hierarchical search to rule out infeasible solutions under kinematic and dynamic constraints for a given trajectory. To leverage also on the potential of unconventional kinematics, the present article follows evolutionary strategies based on genetic algorithms (GA), which have proven to be effective in finding the optimal robot morphology for a given task [12]–[18].

In the presented work we use a two-stage hybrid discrete-continuous optimization scheme for reconfigurable modular robots. The first stage finds the optimal robot design as a

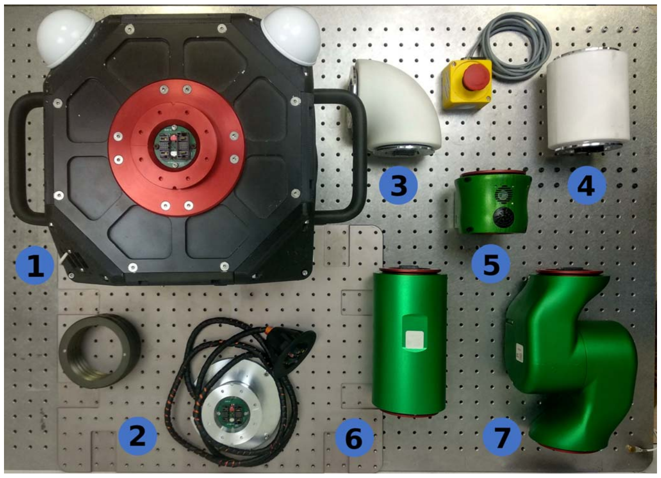


Fig. 2: The components of the reconfigurable robot include: (1) control box, (2) mounting base, (3) elbow Link module, (4) straight Link module, (5) tool-exchanger module, (6) straight Joint module and (7) elbow Joint module.

combination of morphology and base placement within the discrete space of all feasible robot designs. The second stage finds the optimal robot posture and motion in the continuous joint space of this optimal design. Two-stage schemes like this are popular in literature and have been investigated to solve similar problems for example in [15], [19] and [20]. With respect to said papers in this work, we first add the base location as a variable to optimize. Second, we don't restrict the search to robots with a predefined number of DOFs. Third, we make a sensitivity analysis of the effects of the constraints on the final solution. Fourth, after the simulation study, we validate the results with real-world experiments.

This work extends the author's previous work [21] with the following five extensions. First, a genetic algorithm (GA) replaces the brute-force search algorithm to improve scalability of the method with respect to increasing module variety in the robot design optimization stage. Second, the variety of actively actuated modules is expanded by passive linkages, increasing the size of the search space. Third, the added second optimization stage ensures optimal task execution with the optimally designed robot. Fourth, the new scheme accounts for constraints such as the minimum number of degrees of freedom required for the task, the available stock of modules, and each module's individual load capacity. Fifth, the present work further incorporates avoidance of robot self-collisions and robot-environment collisions.

The next section introduces the in-house developed reconfigurable robot prototype used to verify the results of the proposed optimization scheme in real-world experiments. Sec. III details the proposed two-stage optimization scheme in depth. Optimization results are provided and discussed in Sec. IV. The experimental verification of these results in Sec. V proofs the practical effectiveness of the proposed scheme. The article concludes with an outlook in Sec. VI.

II. The Reconfigurable Robotic System

The proposed generative optimization scheme is tested on the in-house developed reconfigurable robot prototype depicted in Fig. 2. It features algorithms for automatic plug & work module discovery, kinematic topology identification, mathematical model generation and software/control configuration as detailed in [22]. This hardware system enables the assembly and experimental study of the found (sub-)optimal solutions in the time-scale of minutes.

A. Module Description

The prototype kit consists of a number of different modules. Every module comprises at least one flange interface for mechanical, electrical and EtherCAT intra-module connectivity. The two types of actuated Joint modules are driven by an "orange" type integrated actuator described in [23], [24] with strain-wave gear of transmission ratio 160:1 and integrated torque sensing. The "straight" type Joint modules are actuated about the common central normal of the modular flange interfaces, while the "elbow" type Joint modules rotate about a central axis orthogonal to the common normal of the modular flange interfaces. For the purpose of this article, two additional passive Link modules have been devised: a straight and an elbow-type version. The mounting base module serves to mechanically fix the robot in the workspace. The end-effector module is a 3D-printed part to simulate the peg-in-hole application used as the task scenario in this article. The control box module accommodates the power supply and a computational unit for the automated morphology recognition and real-time control of the robot.

B. Module Coordinate Frame Assignment

We place a separate coordinate frame in the center of each flange interface as well as on each actuated joint axis. An additional frame is placed in each module's center of mass (CoM). The CoM frame is oriented in parallel to the frame of the "upstream" flange located in direction towards the mounting base module. This convention permits to unequivocally derive the robot physical model (kinematics and dynamics). The respective geometric and inertial module parameters are stored in a centralized module database and collected in the data structure $\boldsymbol{\Phi}$. In Fig. 3, example I shows the frame convention when two Joint modules are connected to each other. Example II shows the frame convention applied to the control box module.

C. Kinematic and Dynamic Algorithms

We follow the spatial algebra notation described in [25] based on the implementation in [26]. We denote the vectors of joint-space positions, velocities and accelerations by $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$. For the remainder of this article, the most relevant dynamics algorithms is the Inverse Dynamics or Recursive Newton-Euler Algorithm (RNEA):

$$\boldsymbol{\tau} = RNEA(\boldsymbol{\Phi}, \boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \; , \qquad (1)$$

with the robot mass matrix $\boldsymbol{M}$ computed using the Composite Rigid Body Algorithm (CRBA) $\boldsymbol{M}(\boldsymbol{q}) = CRBA(\boldsymbol{\Phi}, \boldsymbol{q})$, and the vector $\boldsymbol{h}(\boldsymbol{q}) = RNEA(\boldsymbol{\Phi}, \boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{0})$ holding the gyroscopic and gravitational forces. The gravitational forces can be computed separately as:

$$\boldsymbol{g}(\boldsymbol{q}) = RNEA(\boldsymbol{\Phi}, \boldsymbol{q}, \boldsymbol{0}, \boldsymbol{0}) \; . \qquad (2)$$

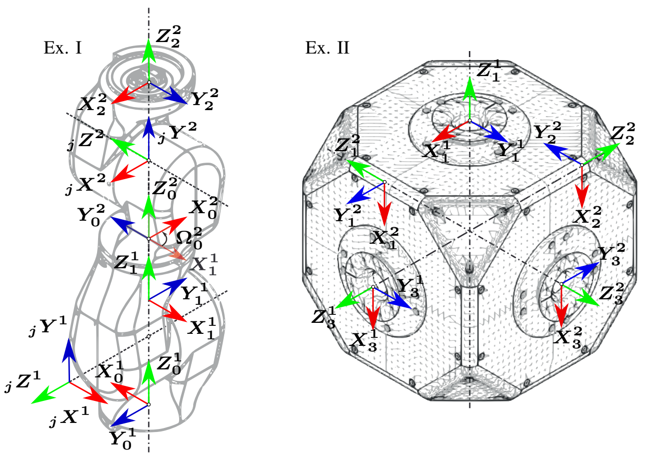


Fig. 3: Example of frames definition for the different type of modules and their connections. The circles indicate frame origins. For better visibility of the frame orientation some coordinate triad are visualized off set. Dashed lines connect the triad to their coordinate origin.

III. Two Stage Optimization Scheme

In this work we use the joint effort required to execute the task as the cost criterion (Sec. III-A). The proposed approach decomposes the hybrid discrete-continuous optimization problem (Sec. III-B) into two stages: a higher level optimization stage (described in Sec. III-D) determining the best design $\mathcal{D}^*$ among the possible combinations of modules and base placement in the discrete robot *design space*, and a lower level stage (described in Sec. III-C) optimizing the cost of a given design by finding the optimum posture $\boldsymbol{q}^*$ in the continuous robot *joint space* $\mathcal{Q} \subset \mathbb{R}^n$. The output of the lower level stage will be used by the higher level stage to compare the *cost* to execute the task by a given design with respect to that of other design candidates.

A. Cost Criterion

In the proposed approach, the cost criterion utilized to evaluate the different configurations of the modular robotic system, is the *effort* [27], [28]:

$$\epsilon(\boldsymbol{\tau}(t)) = \int_0^{t_{tot}} \boldsymbol{\tau}^T(t)\boldsymbol{\tau}(t)dt, \tag{3}$$

where $\boldsymbol{\tau}$ symbolizes the vector of the joint torques. With a discretization of the task trajectory in ${}^jN_p$ points, the effort required for the execution of the j-th task T_j can be approximated in a quasi-static fashion by considering a set of Cartesian poses $\boldsymbol{H}_{ee}(\boldsymbol{q_i})$, with $\boldsymbol{q_i}$ representing the i-th posture, $i = 1, .., {}^j N_p$. The effort (3) is then approximated as follows:

$$\epsilon_{T_j}(\boldsymbol{\tau}_s(T_j)) = \textstyle\sum_{i=1}^{{}^jN_p} (\boldsymbol{\tau}_s^T(\boldsymbol{q_i})\boldsymbol{\tau}_s(\boldsymbol{q_i}))\, \Delta t_i, \tag{4}$$

where Δt_i is the fixed or varying discrete time step associated with the i-th posture $\boldsymbol{q_i}$ and $\sum_{i=1}^{{}^jN_p} \Delta t_i = t_{T_j}$ is the total time to complete the task. As control input we consider only the static torques $\boldsymbol{\tau}_s = \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{F}_{ext}$, where $\boldsymbol{J}(\boldsymbol{q})$ is the manipulator Jacobian and $\boldsymbol{F}_{ext}$ the external force applied by the environment. Considering only the static cost in the energy expenditure for a robot may seem a strong simplification in general. However, for the moderately dynamic tasks analyzed in this work, the static torques are prevalent, so that this approach yields a useful approximation to drive the optimization towards energy-economic solutions.

In the case where not only one, but a set of tasks need to be executed by the robot, the effort required by all distinct tasks can be minimized by considering a multi-objective cost function. If N_t different tasks are to be executed by the robot and each of them is repeated γ_j times, the effort related to the whole process can be computed as $\epsilon_{tot} = \sum_{j=1}^{N_t} \gamma_j\, \epsilon_{T_j}$.

B. The hybrid discrete-continuous optimization problem

A robot design $\mathcal{D} = (\boldsymbol{H}_b, \mathcal{M})$, that includes the robot morphology $\mathcal{M}$ and the coordinate transformation $\boldsymbol{H}_b$ of the robot base link, can then be extracted to minimize all N_t task objectives, where γ_j acts as a weight for the j-th task. The optimization problem to solve becomes:

$$\begin{aligned}
\min_{\mathcal{M}, \boldsymbol{H}_b, \boldsymbol{q}_i} \quad & \epsilon_{tot} = \sum_{j=1}^{N_t} \gamma_j \sum_{i=1}^{{}^jN_p} \Delta t_i (\boldsymbol{\tau}^T(\boldsymbol{q_i})\boldsymbol{\tau}(\boldsymbol{q_i})) \\
\text{s.t.} \quad & \boldsymbol{H}_{ee}(\mathcal{M}, \boldsymbol{H}_b, T_j, \boldsymbol{q_i}) = \bar{\boldsymbol{H}}_{ee}^i \\
& \alpha(\mathcal{M}, \boldsymbol{H}_b, T_j, \boldsymbol{q_i}) > \bar{\alpha}_{th} \\
& \mathcal{M} \in \mathcal{M}_{feasible} \qquad i = 1, ..., {}^j N_p \\
& \boldsymbol{H}_b \in \boldsymbol{H}_{feasible} \qquad j = 1, ..., N_t \\
& \boldsymbol{q}_i \in \mathcal{Q},
\end{aligned} \tag{5}$$

where $\mathcal{M}_{feasible}$ and $\boldsymbol{H}_{feasible}$ are the sets of feasible morphologies and base poses respecting the design constraints, while $\mathcal{Q}$ represents the feasible joint space respecting the joint limits and preventing from collisions. The equality constraint guarantees the reachability of the given poses $\bar{\boldsymbol{H}}_{ee}^i$ defined by the N_t tasks; while the inequality constraint enforces a lower limit $\bar{\alpha}_{th}$ on the force transmission ratio α along a certain direction.

C. Joint space optimization

Given a robot design $\tilde{\mathcal{D}} = (\tilde{\boldsymbol{H}}_b, \tilde{\mathcal{M}})$, for each Cartesian pose $\bar{\boldsymbol{H}}_{ee}^i$ with $i = 1, ..., {}^jN_p$ associated with the discretization of the given task T_j, a solution to the inverse kinematics (IK) minimizing the *effort* must be found. This can be formulated as ${}^jN_p$ optimization problems in the continuous joint coordinates $\mathbf{q_i}$ as follows:

$$\begin{aligned}
\min_{\boldsymbol{q}_i \in \mathcal{Q}} \quad & \epsilon_i = ||\boldsymbol{\tau}_s(\boldsymbol{q_i})||^2 \\
\text{s.t.} \quad & \boldsymbol{H}_{ee}(\tilde{\mathcal{M}}, \tilde{\boldsymbol{H}}_b, T_j, \boldsymbol{q_i}) = \bar{\boldsymbol{H}}_{ee}^i \\
& \alpha(\tilde{\mathcal{M}}, \tilde{\boldsymbol{H}}_b, T_j, \boldsymbol{q_i}) > \bar{\alpha}_{th},
\end{aligned} \tag{6}$$

In this formulation the constraints enforce the position and orientation of the end-effector pose while the objective function minimizes the static torques. If no solution is found for any of the ${}^jN_p$ poses, the task is infeasible. This at the same time ensures the feasibility of the task and that the effort for the whole task ϵ_{T_j} is minimized as well. Constraints on force-transmission ratio α can be set depending on task specifications, in particular if the task requires the robot to apply a force on a certain direction.

1) Implementation: as the objective function defined here does not permit exploiting existing IK solvers, we implemented a customized solver to address (6). The implementation has been done with the NLopt library using the SLSQP algorithm [29], which has been used by Trac-IK [30] to handle well constraints such as joint limits. Nevertheless, it cannot guarantee to find the global minimum, and may result in local solutions – false negatives. These issues can be mitigated by restarting the algorithm with a random seed for several times, as in the Trac-Ik implementation. Moreover, due to the gradient-based nature of this technique, it requires the gradient, which might become troublesome to derive for more complicated constraints. To ensure the posture is feasible to reach without the robot colliding with itself or the environment, a check for collisions is performed by making use of the *PlanningScene* from MoveIt! [31]. If a collision is detected the posture is discarded and a new optimal posture is sought. The robot collision model is derived from the robot URDF, that can be obtained for the modular robot as described in [22] and enriched with an environment model.

2) Orientation constraint relaxation for robots with less than 6 DOFs: Many industrial tasks do not require to control all six Cartesian coordinates. For instance tasks such as welding, painting or "peg-in-hole" of axially symmetric components only require 5 of the 6 DOFs to be controlled, with one of the orientations relaxed. This allow us to evaluate robots with $n <= 6$ DOFs in this study. Openly available IK solver implementations (e.g. , KDL, Trac-IK) require the definition of complete (six dimensional) task-space poses, while the morphologies exhibiting $n < 6$ DOFs cannot fulfill execution of fully-constrained Cartesian poses. To this end, we specify a relaxed orientation constraint on the end-effector pose, so to be able to univocally find solutions for robot with 6 DOFs or less. For the tasks considered in this work which require 5 DOFs, we then set an orientation constraint on the end-effector to be perpendicular to the plane $\mathcal{P}$ where the orientation around that axis is not constrained:

$$h(\boldsymbol{q}) = {}^{w}\hat{\boldsymbol{n}}^{T} \cdot {}^{w}\hat{\boldsymbol{z}}_{ee} = -1, \tag{7}$$

where ${}^{w}\hat{\boldsymbol{n}}$ is the unit vector normal to the plane $\mathcal{P}$ and ${}^{w}\hat{\boldsymbol{z}}_{ee}$ the z-axis of the end-effector frame, both expressed in the global frame $\{w\}$. When working with gradient-based optimization algorithms, the gradient of this constraint is computed as

$$\nabla h(\boldsymbol{q}) = {}^{w}\hat{\boldsymbol{n}}^{T} \cdot \boldsymbol{\Omega}({}^{w}\hat{\boldsymbol{z}}_{ee})^{T} \cdot \boldsymbol{J}(\boldsymbol{q}), \tag{8}$$

by exploiting the relation

$${}^{w}\dot{\hat{\boldsymbol{z}}}_{ee} = \omega \times {}^{w}\hat{\boldsymbol{z}}_{ee} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \times {}^{w}\hat{\boldsymbol{z}}_{ee} = \boldsymbol{\Omega}({}^{w}\hat{\boldsymbol{z}}_{ee})^{T} \cdot \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{9}$$

D. Design space optimization

The higher-level optimization problem can then be solved by using the output of the lower level optimization problems. The remaining optimization problem can be formulated in the discrete robot design variables $\mathcal{M}$ and $\boldsymbol{H_b}$ as:

$$\begin{aligned} \min_{\mathcal{M}, \boldsymbol{H}_b} \quad & \epsilon_{tot} = \sum_{j=1}^{N} \gamma_j \epsilon_{T_j} \\ \text{s.t.} \quad & \mathcal{M} \in \mathcal{M}_{feasible} \qquad 1 \leq j \leq N \\ & \boldsymbol{H}_b \in \boldsymbol{H}_{feasible}, \end{aligned} \tag{10}$$

where the cost function value can be directly computed from the output of the lower-level optimizations ϵ_{T_j} and the number of repetitions of each tasks γ_j. Only discrete variables are left in this problem. Constraints on the robot design can be set to limit the design space to the allowed configurations.

For solving the higher level optimization problem a GA has been implemented for its simplicity and straight-forward analogy to the physical assembly of modular robotic structures. In fact operations such as the *crossover*, which swaps two sub-strings (or schema) from two different genotypes is equivalent to swapping two sub-assembly patterns from two robot assemblies; or as a *mutation*, which randomly flips a bit in a gene encoding, is equivalent to replace a robot module with another.

1) Encoding: With our robot modules and the conventions outlined in Sec. II, we employ an integer alphabet for encoding all kinematic and dynamic properties of the module. For the problem addressed, it follows the "principle of meaningful building blocks" [32] and produces shorter and more meaningful building blocks than binary-based alphabets. For example, a combination of one straight and one elbow Joint module (a good building block for many robots) can be encoded with just two integers (e.g. the string "1 2"). This makes the encoding robust as it provides a schema that is difficult to break by crossover operations and with low probability to be mutated, therefore likely to be propagated also to future generations.

The set of s different module types can be completely encoded by making use of an integer alphabet of cardinality s. A string of r integers univocally determine the morphology $\mathcal{M}$ composed of r modules. As the morphology may comprise active Joint and passive Link modules in the same way, we can have $r \geq n$ modules in the chain. This entails the morphology string to generally vary in length. The integer encoding can be expanded to include the six parameters defining the transformation $\boldsymbol{H}_b$ from the global frame to the robot's base frame: d_x, d_y, d_z for the position coordinates and d_ν, d_μ, d_ϵ for the relative orientation in the representation of choice (Euler angles, quaternions etc.). They are discretized to the desired granularity and prepended as sub-string. The length of the total string $\mathcal{D}$ completely describing the robot design is then the $(6+r)$-integer string:

$$\mathcal{D} = \text{"}d_x\, d_y\, d_z\, d_\nu\, d_\mu\, d_\epsilon\, d_1\, d_2\, \ldots\, d_r\text{"}\ . \tag{11}$$

Note that we exclude the end-effector module. It must be present in every design, and would carry no information in the encoding string. It is added afterwards to any generated design for the cost evaluation.

2) Decoding: The GA *evaluation* function can decode the string and univocally reconstruct the robot model from it, by considering the first six integers for $\boldsymbol{H}_b$ and the last r genes to obtain $\mathcal{M}$. Thanks to the convention established in Sec. II the r modules can be iteratively added to reconstruct the full kinematic and dynamic model. For each considered task T_j, the lower level optimization algorithm is called a total of ${}^{j}N_p$ times to compute the overall cost ϵ_{tot} of the current design $\mathcal{D} = (\boldsymbol{H}_b, \mathcal{M})$.

3) Design constraints handling: Constraints on the robot design can be handled at this point to enforce $\mathcal{M} \in \mathcal{M}_{feasible}$ and $\boldsymbol{H}_b \in \boldsymbol{H}_{feasible}$. Examples of constraints considered are:

- Constraints on the maximum number of available modules of a certain type to prevent obtaining an optimal robot that cannot be realized.
- Constraints on minimum number of DOFs. Operations as mutation might create genotypes with too many passive links and not enough DOFs. These could be discarded to avoid issues with the evaluation function.
- Constraints on maximum weight a module can support due to structural limits. These can be tackled by knowing the weight of every successive module in the string.
- Constraints on the base placement, given by possible obstacles present in the workcell. The input to this could be a CAD model or data from a camera.

IV. Optimization Results

The proposed method is utilized to optimize for a peg-in-hole task of a 2×2 pattern on a vertical plane passing through the x and z axes of the global frame, where the four holes are at the corners of a square of 12 cm sides, which is centered at [0.53, 0.0, 0.18] w.r.t. the global frame. This can be a challenging task for robots with less than six DOFs, while an optimal topology of the Link modules for modular robots can address this problem. The evaluated scenario is defined by a work-cell where the base of the robot can be placed on a 1.2 m×0.6 m table with pre-made mounting holes in a way that it allows for a discretization with steps of 0.2 m. It also permits to consider the angle offset ϕ with respect to the z-axis of the global frame with steps of 90°, which is reduced to the two values $\{0.0, 1.57\}$ due to the symmetry of the modules. The base pose is encoded by two integers $d_x \in \mathcal{X} = \{0, 1, .., 7\}$ and $d_y \in \mathcal{Y} = \{0, 1, 2, 3\}$ for the position and one integer $d_\phi \in \mathcal{B} = \{0, 1\}$ for the orientation. This results in the $(3+r)$-integer string $\mathcal{D} = d_\phi d_x d_y d_1 d_2 ... d_r$, where the first three integers determine the transform from the global frame to the base frame of the robot $\boldsymbol{H}_b$. For what regards the robot morphology $\mathcal{M}$, s=4 types of modules are considered, where the representation of each module can be encoded by an integer $d_i \in \mathcal{A} = \{0, 1, 2, 3\}$. The encoding is 0 for the straight Link, 1 for the elbow Joint, 2 for the straight Joint and 3 for the elbow Link module.

To successfully perform the task also with robots with $n < 6$, we implemented the relaxation of the orientation constraint associated with the corresponding normal plane $\mathcal{P}$, as explained in Sec. III-C, rather than specifying a full

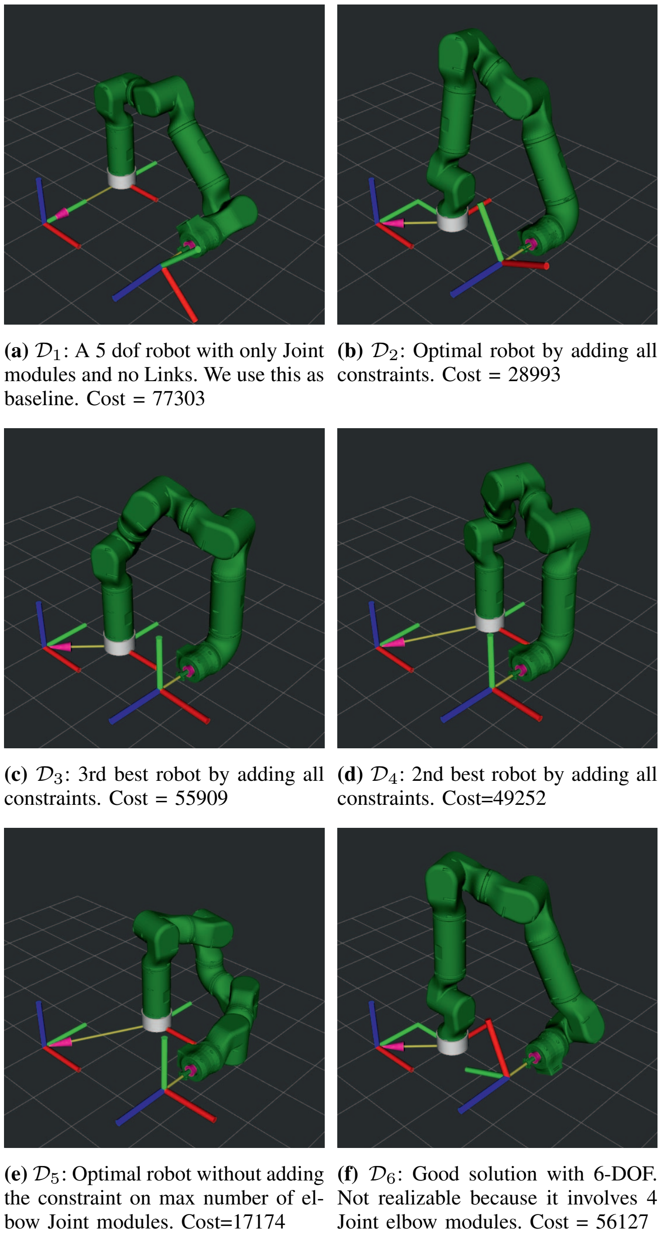

(a) $\mathcal{D}_1$: A 5 dof robot with only Joint modules and no Links. We use this as baseline. Cost = 77303
(b) $\mathcal{D}_2$: Optimal robot by adding all constraints. Cost = 28993
(c) $\mathcal{D}_3$: 3rd best robot by adding all constraints. Cost = 55909
(d) $\mathcal{D}_4$: 2nd best robot by adding all constraints. Cost=49252
(e) $\mathcal{D}_5$: Optimal robot without adding the constraint on max number of elbow Joint modules. Cost=17174
(f) $\mathcal{D}_6$: Good solution with 6-DOF. Not realizable because it involves 4 Joint elbow modules. Cost = 56127

Fig. 4: The best results obtained in simulation by optimizing for the “peg-in-hole“ task

orientation for the end-effector. The task has been approximately discretized in eight Cartesian poses: four pre-insertion poses and four insertion poses. Without loss of generality, we consider all poses in the task to have the same time duration Δt_i = 5.0 s so that the total time to complete the task would be of $t_{T_j} = 40.0$ s. A constraint on the lower bound of the force-transmission ratio along the insertion direction $\bar{\alpha}_{th}$ of 0.2 is also set, whose value was set heuristically. Joint limits are set as 2.0 rad for the elbow Joint modules and 2.8 rad for the straight Joint modules. Constraints at the design space level are also enforced:

- The max number of elbow or straight Joint modules that can be used in the robot design is three for each. For the elbow and straight Link modules this is 1 each. This is set according to the actual availability of modules.
- The minimum number of DOFs the robot can have is

five, as the task will not be feasible with less DOFs.

- The maximum weight the Link module can support is set to 6 Kg. This is set according to the specifications of the structural limits of the prototype based on data provided by the analysis of the prototype CAD model. We can discard designs where the sum of the weight of the successive modules in the chain surpass the limit. No limit is set for the Joint modules.
- We consider the work-cell table to be occupied for $x > 0.6\,\mathrm{m}$, thereby restricting our design space search to a square of $0.6 \times 0.6\,\mathrm{m}^2$.

The implementation of the GA is done by making use of the DEAP toolbox, where the mutation and crossover operations are modified in such a way that they generate designs only using the correct alphabet for every gene d_i and avoid creating unfeasible designs. A population size of 40 individuals was used, which are evolved over time for 80 generations, with a crossover probability of 0.5 and a mutation probability of 0.1. The result of the simulation is shown in Fig. 5 where the evolution of the cost over the generations is shown. The average size (r) of the robot structure sub-string is also plotted. While we cannot present all the solutions here, due to the limited space, a number of the best results, found in the "hall of fame" of evaluated individuals, are reported below:

- $\mathcal{D}_1 = (\boldsymbol{H}_b^1, \mathcal{M}_1) =$ "$0\,0\,2\,2\,1\,1\,2\,1$". A generic 5-DOF robot, without adding any Link modules. This is a versatile robot that can be used for many tasks and what we may naively build without running the task-based optimization. We consider this as a baseline against the task-specific robots selected by the optimization. The total cost for this design is $\epsilon_{tot} = 77303$.
- $\mathcal{D}_2 = (\boldsymbol{H}_b^2, \mathcal{M}_2) =$ "$1\,1\,1\,1\,2\,1\,1\,2\,3$". The best robot design when adding all constraints. Having the elbow joint as the 1st joint is certainly not an intuitive solution, which however performs particularly well for this task. The total cost is $\epsilon_{tot} = 28993$.
- $\mathcal{D}_4 = (\boldsymbol{H}_b^4, \mathcal{M}_4) =$ "$0\,1\,2\,2\,1\,1\,1\,2\,3$". It is the 2nd classified design which has a total cost of $\epsilon_{tot} = 49252$.
- $\mathcal{D}_3 = (\boldsymbol{H}_b^3, \mathcal{M}_3) =$ "$0\,1\,1\,2\,1\,1\,1\,2\,3$". It is the 3rd classified design and the "brother" of $\mathcal{D}_4$. The robot design is the same as $\mathcal{M}_3 = \mathcal{M}_4$, with a different base position. Comparison with $\mathcal{D}_4$ shows the effect of the base position on the final cost: $\epsilon_{tot} = 55909$.

To evaluate the effect of current limitation on the number of available modules, we execute again the optimization without enforcing this constraint. The major results are as follows:

- $\mathcal{D}_5 = (\boldsymbol{H}_b^5, \mathcal{M}_5) =$ "$0\,1\,2\,2\,1\,1\,3\,1\,1$". Despite being practically unrealizable (requiring four elbow Joint modules), this design is the most optimal solution as the final cost is very low due to the particular structure of the robot, especially when close to the working design. The total cost is $\epsilon_{tot} = 17174$.
- $\mathcal{D}_6 = (\boldsymbol{H}_b^6, \mathcal{M}_6) =$ "$1\,1\,1\,1\,2\,1\,1\,2\,1$". This design also requires four elbow Joints and is therefore unrealizable. It possesses six DOFs, allowing for a larger variety of tasks compared to 5-DOF robots, while rendering a comparable cost as some of the above-mentioned optimal designs. The total cost is $\epsilon_{tot} = 56127$.

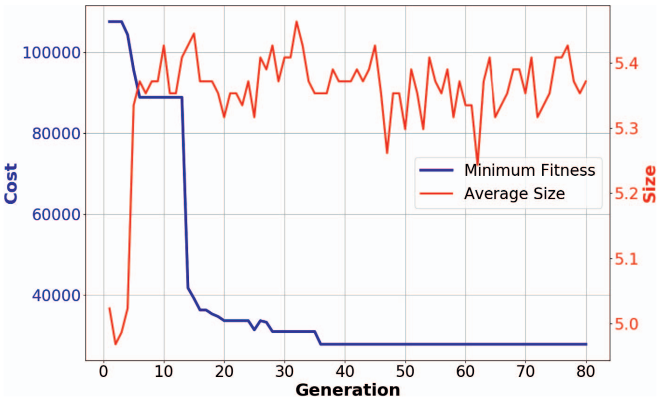


Fig. 5: Cost evolution over generations and average number of robot DOFs over generations.

Performance-wise the two-stage optimization scheme proposed in this work drastically reduces the computation time with respect to the brute-force approach used in [21]. For optimization runs, which required multiple days to complete, we went down to execution times in the range of one to three hours. At the same time we have expanded the search space by adding more modules. The results were obtained on a 3.4 GHz AMD Ryzen 7 1700x 8-core processor.

V. Experiments

After the optimization phase, the cost estimates are evaluated by performing the task with real robot prototype designs. Therefore, the actual values for the cost criterion are computed from the joint torque measurements recorded during each experiment. The experimentally obtained values are referenced against their simulated counterparts.

Due to the practical limitation of available modules stock, only the first 4 designs $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$ and $\mathcal{D}_4$ of the group shown in Sec. IV are assembled and tested with the peg-in-hole task. The task is executed using a Cartesian impedance controller as the specification and tuning of the stiffness values in the task frame of the peg-in-hole task is more straightforward compared to the specification in joint space. The relative location between the robot and the targeted holes are programmed through kinesthetic teaching. The reference poses $\bar{\boldsymbol{H}}_{ee}^i$ are obtained from forward kinematics starting from the optimized postures $\boldsymbol{q}_i^*$.

To compare the four different designs, Fig. 6 shows the instantaneous power of the actual control input $P_\tau(t) = \boldsymbol{\tau}^T(t)\boldsymbol{\tau}(t)$. A scaled-up Cartesian y position reference is also plotted to relate the costs with the motion phase of the peg. The total control effort for the whole task ϵ_{tot}, which is proportional to the energy spent, can be computed by integrating power over time. Fig. 7 depicts the corresponding efforts. The effort computed by integrating the instantaneous power of $\boldsymbol{\tau}(t)$ is shown in blue. The effort corresponding to $\boldsymbol{g}(t)$ obtained from the experimental torque measurements is shown in red. The effort derived from simulation by using (4) is illustrated in yellow. From Fig. 7 it is apparent, that the

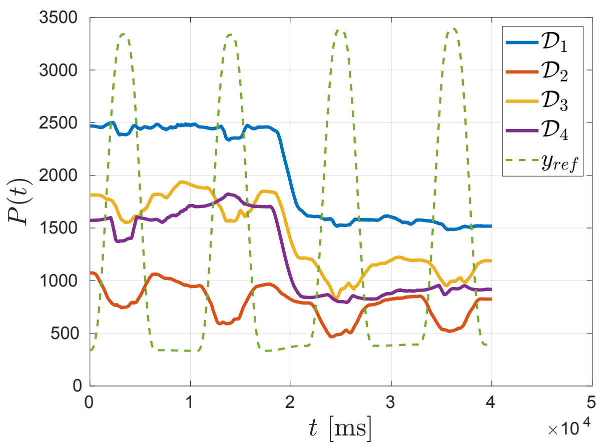


Fig. 6: The instantaneous power of the control input during the task execution for the 4 evaluated robot designs over time

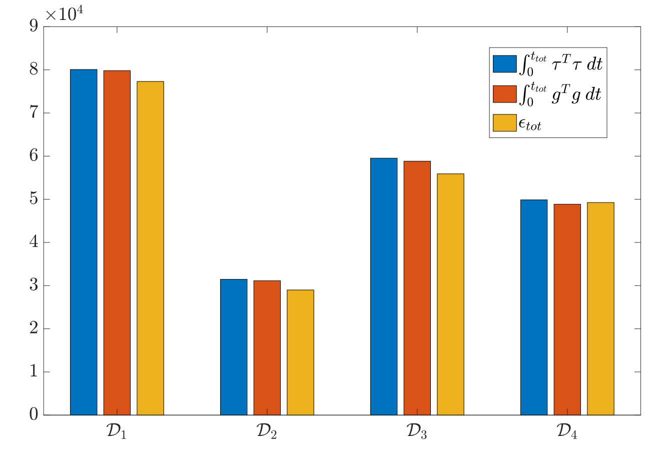


Fig. 7: The total effort for the 4 designs tested. In blue computed using real torque measurements $\boldsymbol{\tau}(t)$, in red using the gravity vector $\boldsymbol{g}(t)$, in yellow the value derived from the simulation using (4)

approximation of the actual effort based on the static torques at certain waypoints, closely matches the experimental data. It is sufficiently accurate to practically predict the minimum effort design for this low speed executed task. The found optimal robot design $\mathcal{D}_2$ demonstrates an energy demand reduction of about 62 % compared to our baseline robot design $\mathcal{D}_1$. The designs $\mathcal{D}_3$ and $\mathcal{D}_4$ also exhibit clearly visible energy demand reductions. Moreover it can be observed how much design $\mathcal{D}_4$ with its base placed at $x = 0.2$, $y = 0.4$ improves over design $\mathcal{D}_3$ with its base placed at $x = 0.2$, $y = 0.2$. This underlines how the same robot morphology $\mathcal{M}_3 = \mathcal{M}_4$ may demonstrate different energy consumption, if the position of its base varies. The marginal discrepancies between the simulated and actual final cost can be explained by the errors introduced by factors such as:

- the discretization of the task in ${}^jN_p$ static postures and the cost approximation based on static torques,
- the use of an impedance Cartesian controller to execute the task, such that compliance causes the actual joint positions to be slightly different from the ones computed in simulation.
- possible offsets in the torque readings, which accumulate in the cost criterion via integration over time.

Finally, we study the sensitivity of the final cost and energy consumption of the robot designs with respect to the lower level robot posture optimization. For instance, fixing the robot design and selecting slightly different orientations of the end-effector can lead to significant changes in the posture and the final result. For this reason we show the effect of the relaxation of constraint (7) from an equality to an inequality constraint of the kind:

$$g(\boldsymbol{q}) = {}^w\hat{\boldsymbol{n}}^T \cdot {}^w\hat{\boldsymbol{z}}_{ee} \leq -0.996, \qquad (12)$$

which allows for orientation errors of about 5°. For tasks where loose tolerances like these are acceptable, the optimization is able to select a better posture with a lower effort cost. The cost for the two different constraints is compared in two experiments with robot design $\mathcal{D}_3$. Fig. 8 shows the instantaneous power during the task execution for both experiments. The integration to compute the total effort

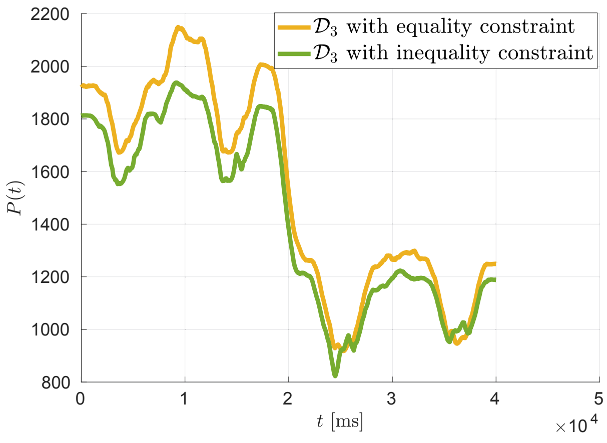
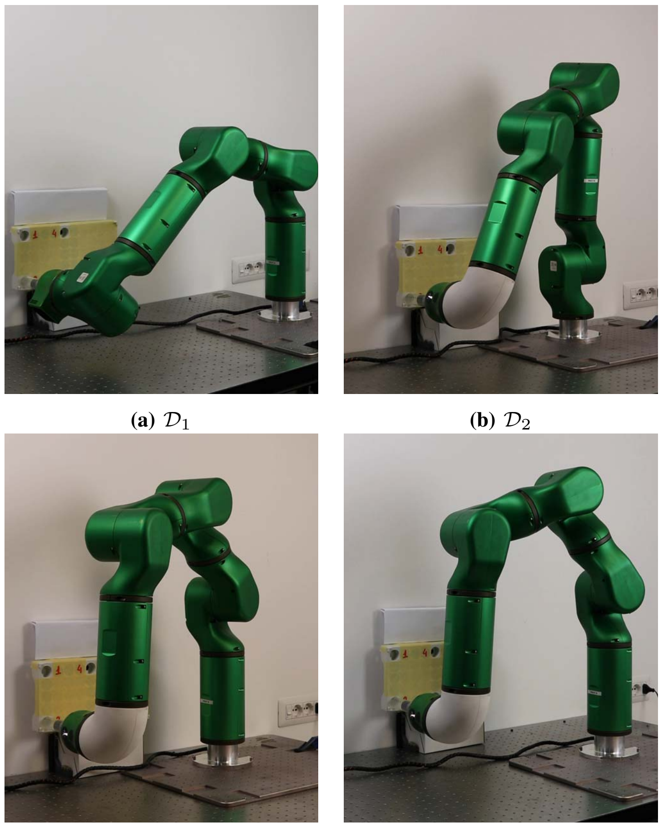


Fig. 8: The effect of joint space optimization on the instantaneous power of $\boldsymbol{\tau}(t)$ for a fixed design $\mathcal{D}_3$. In yellow if the orientation constraint is expressed as an equality constraint, in green if expressed as an inequality constraint to allow for a tolerance of 5°

yields a reduction in the cost obtained with the original constraint $\epsilon_{tot} = 55909$ to a cost of $\epsilon_{tot} = 44775$ with the relaxed orientation constraint. It's clear that this can be exploited for tasks, which allow loose tolerances on the orientation, or without relaxing the constraint with robots that have a null-space associated to the task.

VI. Conclusions

This paper presented a study and analysis on how the design of reconfigurable robots can be optimized for a given task to provide enhanced benefits and performance. A two stage optimization scheme to generate optimal fit-to-task assemblies was introduced and demonstrated for the minimum effort execution of a sequence of peg-in-hole tasks mimicking an assembly scenario in a confined environment. The derived optimal and multiple sub-optimal assemblies were realized using a reconfigurable robot prototype and evaluated in peg-in-hole experiments. The cost predicted by the models generated for each assembly are in good agreement with that obtained from the real experiments. A sensitivity study of the cost w.r.t. to a relaxation in



(a) $\mathcal{D}_1$ **(b)** $\mathcal{D}_2$

(c) $\mathcal{D}_3$ **(d)** $\mathcal{D}_4$

Fig. 9: The real-world assembly of the best robots performing the "peg-in-hole" task

orientation constraints provides interesting insights to further task-specific robot optimization potentials.

Beyond adopting an adaptive search space discretization scheme, future work will concentrate in two directions. We will target to scale the presented scheme to a larger variety of modules, multiple collaborating robots as well as in more complex task scenarios. Currently a full optimization run takes between one and three hours, making it usable to optimally re-purpose reconfigurable robot cell in response to market fluctuations with a forewarning of that timescale. To improve flexibility, the second direction hence aims at further reducing the time required to derive the optimal design solution, making it close to the timescale needed to assemble and program the reconfigurable robot hardware.

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 101016007 CONCERT.

References

[1] Y. Koren, *The global manufacturing revolution: product-process-business integration and reconfigurable systems*. John Wiley & Sons, 2010, vol. 80.

[2] N. Brahimi, A. Dolgui, E. Gurevsky, and A. R. Yelles-Chaouche, "A literature review of optimization problems for reconfigurable manufacturing systems," *IFAC-PapersOnLine*, vol. 52, no. 13, pp. 433–438, 2019.

[3] A. R. Yelles-Chaouche, E. Gurevsky, N. Brahimi, and A. Dolgui, "Reconfigurable manufacturing systems from an optimisation perspective: A focused review of literature," *Int. J. Prod. Res.*, pp. 1–19, Oct. 2020.

[4] L. E. Kavraki and S. M. LaValle, "Motion planning," in *Springer Handbook of Robotics*. Springer, 2016, pp. 139–162.

[5] C. Paredis and P. K. Khosla, "An approach for mapping kinematic task specifications into a manipulator design," 1991.

[6] D. G. Luenberger and D. G. Luenberger, "Linear and nonlinear programming," 2016.

[7] S. Shirafuji and J. Ota, "Kinematic synthesis of a serial robotic manipulator by using generalized differential inverse kinematics," *IEEE Trans. Robot.*, vol. 35, no. 4, pp. 1047–1054, 2019.

[8] T. Campos de Almeida, S. Marri, and H. Kress-Gazit, "Automated synthesis of modular manipulators' structure and control for continuous tasks around obstacles," *Robotics: Science and Systems*, 2020.

[9] T. Fukuda and S. Nakagawa, "Dynamically reconfigurable robotic system," in *IEEE Int. Conf. Robot. Autom.*, 1988, pp. 1581–1586.

[10] S. Ha, S. Coros, A. Alspach, J. M. Bern, J. Kim, and K. Yamane, "Computational design of robotic devices from high-level motion specifications," *Proc. IEEE Trans. Robot.*, vol. 34, no. 5, pp. 1240–1251, 2018.

[11] S. B. Liu and M. Althoff, "Optimizing performance in automation through modular robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 4044–4050.

[12] D. Pathak, C. Lu, T. Darrell, P. Isola, and A. A. Efros, "Learning to control self-assembling morphologies: a study of generalization via modularity," *arXiv preprint arXiv:1902.05546*, 2019.

[13] C. Leger *et al.*, *Automated synthesis and optimization of robot configurations: an evolutionary approach*. CMU, USA, 1999.

[14] Z. Bi and W.-J. Zhang, "Concurrent optimal design of modular robotic configuration," *J. Robot. Syst.*, vol. 18, no. 2, pp. 77–87, 2001.

[15] O. Chocron and P. Bidaud, "Genetic design of 3d modular manipulators," in *IEEE Int. Conf. Robot. Autom.*, vol. 1, 1997, pp. 223–228.

[16] J.-O. Kim and P. K. Khosla, "Design of space shuttle tile servicing robot: an application of task based kinematic design," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1993, pp. 867–874.

[17] I.-M. Chen and J. W. Burdick, "Determining task optimal modular robot assembly configurations," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 1, 1995, pp. 132–137.

[18] E. Icer, H. Hassan, K. El-Ayat, and M. Althoff, "Evolutionary cost-optimal composition synthesis of modular robots considering a given task," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 3562–3568.

[19] W. K. Chung, J. Han, Y. Youm, and S. Kim, "Task based design of modular robot manipulator using efficient genetic algorithm," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 1, 1997, pp. 507–512.

[20] S. Ha, S. Coros, A. Alspach, J. Kim, and K. Yamane, "Task-based limb optimization for legged robots," in *IEEE/RSJ Int. Conf. Intell. Robot. System.*, 2016, pp. 2062–2068.

[21] E. Romiti, N. Kashiri, J. Malzahn, and N. Tsagarakis, "Minimum-effort task-based design optimization of modular reconfigurable robots," in *2021 IEEE Int. Conf. Rob. Autom.*, 2021, pp. 9891–9897.

[22] E. Romiti, J. Malzahn, N. Kashiri, F. Iacobelli, M. Ruzzon, A. Laurenzi, E. Mingo Hoffman, L. Muratore, *et al.*, "Toward a plug-and-work reconfigurable cobot," *IEEE/ASME Trans. Mech.*, vol. 27, no. 5, pp. 3219–3231, 2022.

[23] N. Kashiri, L. Baccelliere, L. Muratore, A. Laurenzi, *et al.*, "Centauro: A hybrid locomotion and high power resilient manipulation platform," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 1595–1602, 2019.

[24] "Alberobotics," https://alberobotics.it/, accessed: 2023-02-30.

[25] R. Featherstone, *Rigid Body Dynamics Algorithms*, 2008, vol. 136, no. 1. [Online]. Available: http://link.springer.com/10.1007/978-1-4899-7560-7

[26] M. L. Felis, "RBDL: an efficient rigid-body dynamics library using recursive algorithms," *Aut. Robots*, vol. 41, no. 2, pp. 495–511, 2017.

[27] D. E. Kirk, *Optimal Control Theory*. Prentice-Hall, 1970.

[28] B. J. Martin and J. E. Bobrow, "Minimum-effort motions for open-chain manipulators with task-dependent end-effector constraints," *Int. J. Robot. Res.*, vol. 18, no. 2, pp. 213–224, 1999.

[29] D. Kraft *et al.*, "A software package for sequential quadratic programming," 1988.

[30] P. Beeson and B. Ames, "Trac-ik: An open-source library for improved solving of generic inverse kinematics," in *IEEE-RAS Int. Conf. Human. Robot.*, 2015, pp. 928–935.

[31] S. Chitta, I. Sucan, and S. Cousins, "Moveit![ros topics]," *IEEE Robot. Autom. Mag.*, vol. 19, no. 1, pp. 18–19, 2012.

[32] D. E. Goldberg, *Genetic algorithms in Search, Optimization, and Machine Learning*. Addison Wesley Publishing Co. Inc., 1989.