2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Optimizing performance in automation through modular robots

Stefan B. Liu and Matthias Althoff
Department of Informatics, Technical University of Munich, Germany, Email: [stefan.liu | althoff]@tum.de

***Abstract*— Flexible manufacturing and automation require robots that can be adapted to changing tasks. We propose to use modular robots that are customized from given modules for a specific task. This work presents an algorithm for proposing a module composition that is optimal with respect to performance metrics such as cycle time and energy efficiency, while considering kinematic, dynamic, and obstacle constraints. Tasks are defined as trajectories in Cartesian space, as a list of poses for the robot to reach as fast as possible, or as dexterity in a desired workspace. In a simulated comparison with commercially available industrial robots, we demonstrate the superiority of our approach in randomly generated tasks with respect to the chosen performance metrics. We use our modular robot *proModular.1* for the comparison.**

Fig. 1. *proModular.1* is a set of modules (left) that can be composed into an industrial robot (right).

I. Introduction

Increasing demand for mass-customization requires robots that can adapt to frequently changing tasks. However, because standard industrial robots cannot change their kinematics, it may occur that new positions are unreachable, or new trajectories can only be followed in a suboptimal manner. Modular robots – robots which are composed from a certain number of reconfigurable hardware modules – are a potential solution [1]. Conceptual advantages of modular robots are imminent: flexible reconfiguration allows the robot to adapt to changing tasks; broken robots can be easily repaired by exchanging affected modules; and, by mass-producing standard modules, the costs of robots can be reduced.

Nowadays, a robot is typically selected for a specific application according to the required performance [2], e.g., weight-carrying capacity, working envelope, repeatability, and speed. Due to increasing efforts of reducing carbon footprints, the energy consumption of robots is also becoming a relevant factor [3]. Further performance metrics of manipulators have been surveyed in [4]. In this work, we propose a computationally-feasible automatic module selection process that optimizes these performance metrics. We compare the performance for several manufacturing applications categorized by their predominant path type (see Tab. I): 1) fully specified trajectories, e.g., dispensing a fluid on a fixed part at a steady speed; 2) a list of poses for the robot to reach as fast as possible, e.g., handling a part; 3) optimizing the designated working area of the robot using workspace performance metrics (see Tab. II) if the path is not known. In all cases, the composed modular robot shall be capable of reaching the desired poses, holding its own weight, exerting the required external forces, and avoiding collisions with itself and surrounding objects. We do not regard repeatability in this work, because it depends largely on the mechanical structure and the controller, which are not part our evaluation.

TABLE I
Typical applications of robots manipulators in manufacturing (from [2])

Application	Predominant path type	Forces
Welding	trajectory in cart. space	low
Dispensing	trajectory in cart. space	low
Processing	trajectory in cart. space	high
Handling	list of poses	mixed
Machine tooling	list of poses	mixed
Assembly	list of poses	mixed

To evaluate performance gains of modular robots, we have designed *proModular.1*, a set of 52 reconfigurable modules for industrial use (see Fig. I). These modules are capable of automatically generating the composed kinematic, dynamic, and geometric model given the information stored inside each module. The joint modules can exert torques at a range from 25 to 226 Nm. We evaluate performance gains by these modules for industrial tasks by comparing them to three commercially available industrial robots for 2700 randomly generated tasks, for which we each find an optimal module composition.

Previous works on task-based optimization of module compositions can be found in [1], [5]–[14]. These have dealt with large search-spaces in different manners: 1) using incidence matrices for efficient enumeration [6], [7], 2) using Genetic Algorithms for finding the optimal module sequence [10]–[13], and 3) choosing a hierarchical elemination approach [1], [9], [10]. The authors of [5], [13], [15] also addressed optimizing modular robots for fault tolerance. Most of previous authors only consider optimizing the kinematics, or module masses [8], [10].

In our work we choose a practical approach, which only

978-1-7281-7395-5/20/$31.00 ©2020 IEEE

TABLE II

A SELECTION OF MANIPULATOR PERFORMANCE METRICS (FROM [4])

Requires	Performance metrics
Jacobian	Manipulability, minimum singular value, condition number, isotropy, velocity ratio
Kinematics	Reachability, Dexterity, operating volume, orientation angle workspace
Dynamics	Dynamic manipulability, dynamic isotropy

enumerates compositions resembling standard robot kinematics [16], combined with a hierarchical elimination to exclude infeasible module sequences. In contrast to previous work, we consider the full dynamics of the modular robots. In addition to [1], we also consider collision avoiding trajectories using Rapidly-exploring random trees (RRT) [17] connected with time-optimal trapezoidals [18] to plan around complex obstacles and prevent self-collision. We are able to demonstrate the performance gains of modular robots versus commercially available industrial arms in a simulated comparison.

In Sec. II we formulate the optimization problems for the three previously-mentioned cases in manufacturing. The module selection process is then explained in Sec. III. We evaluate the performance gains in Sec. IV and conclude the paper in Sec. V.

II. Problem Formulation

We aim to find the optimal composition of modules given task specifications in Cartesian space and an obstacle map. Although we only consider energy, cycle time, and dexterity, the performance metrics of Tab. II and [4] can be included without much additional computational burden, because our module selection process already requires simulating the Jacobian, as well as the kinematics and dynamics of the composed robot. Subsequently, we denote the set of possible compositions as $\mathcal{C} = \{C_1, C_2, ..., C_N\}$, where C_i is the i^{th} composition and N is the number of considered compositions. We perform evaluations for the following three cases.

Case 1: Desired trajectory

In applications such as in welding, dispensing, and processing, a desired trajectory of the end-effector is typically specified by the time-dependent functions $p(t), \dot{p}(t), \ddot{p}(t), R(t), \omega(t), \dot{\omega}(t), F(t)$, which are the desired three-dimensional position, velocity, acceleration, the 3×3 rotation matrix, the three-dimensional rotational velocity, acceleration, and the six-dimensional Cartesian force profile, respectively. These functions are used to compute the motor torque vector $u_i(t)$ and the joint velocity vector $\dot{q}_i(t)$, which are different for each composition C_i. We minimize the energy consumption E:

$$\min_{C_i \in \mathcal{C}} E_i, \tag{1}$$

$$E_i = \int_0^{t_f} |u_i(t)|^T |\dot{q}_i(t)| dt \tag{2}$$

subject to $\forall t \in [0, t_f]$

$$\exists q_i(t) \in \left[\underline{q}_i, \overline{q}_i\right] \qquad \text{(solution within joint limits)} \tag{3}$$

$$\exists u_i(t) \in [\underline{u}_i, \overline{u}_i] \qquad \text{(torque limits)} \tag{4}$$

$$\mathcal{A}_i(q_i(t)) \cap \mathcal{O} = \emptyset \qquad \text{(no collisions)} \tag{5}$$

where $\mathcal{A}_i(q_i(t))$ returns the occupancy of the robot (no self-collision allowed) in Cartesian space and $\mathcal{O}$ is the set of obstacles. To compute the motor torque requires evaluating the robot dynamics, e.g., using the Newton-Euler algorithm [19, Ch. 7].

Case 2: List of desired poses

For applications such as handling, machine tooling, and assembly, the exact trajectory is not important, rather than a fast transition between poses which are given as a list of pairs (p, R), and forces F (e.g., a payload to transport) to minimize *cycle time*. We therefore search for the fastest collision-free trajectory between each pose for every composition C_i. The optimization problem now depends on a weighted sum of the time $t_{f,i}$ and energy consumption E_i to reach the final pose, using weighting factors w_t, w_e:

$$\min_{C_i \in \mathcal{C}} w_t t_{f,i} + w_e E_i, \tag{6}$$

$$E_i = \int_0^{t_{\text{f,i}}} |u_i(t)|^T |\dot{q}_i(t)| dt \tag{7}$$

subject to the same constraints of (3)–(5).

Case 3: Desired workspace

Optimizing the performance of the robot within a desired workspace of the end-effector can be done, if the exact motions are not known beforehand (e.g., motions are generated online). As an example, we maximize the mean dexterity index [4] integrated over M uniformly sampled points of a three-dimensional desired workspace:

$$\max_{C_i \in \mathcal{C}} \frac{\sum_{j=1}^{M} D_{i,j}}{M}, \tag{8}$$

$$D_{i,j} = \frac{1}{3}\left(\frac{\Delta\alpha_{i,j}}{2\pi} + \frac{\Delta\beta_{i,j}}{\pi} + \frac{\Delta\gamma_{i,j}}{2\pi}\right) \tag{9}$$

where $\Delta\alpha_{i,j}, \Delta\beta_{i,j}, \Delta\gamma_{i,j}$ are the range of possible yaw, pitch and roll angles about each of the sampled points, evaluated via the existance of an inverse kinematics solution.

III. Optimal module composition

Evaluating the costs introduced in the previous section for all possible module compositions is computationally expensive. Since the number of possible compositions increases exponentially with the length of the module sequence, calculation may become infeasible. In the following, we introduce our contributions towards a computationally feasible optimization: enumeration of common industrial kinematics, their hierarchical elimination, and the generation of collision models of modular robots.

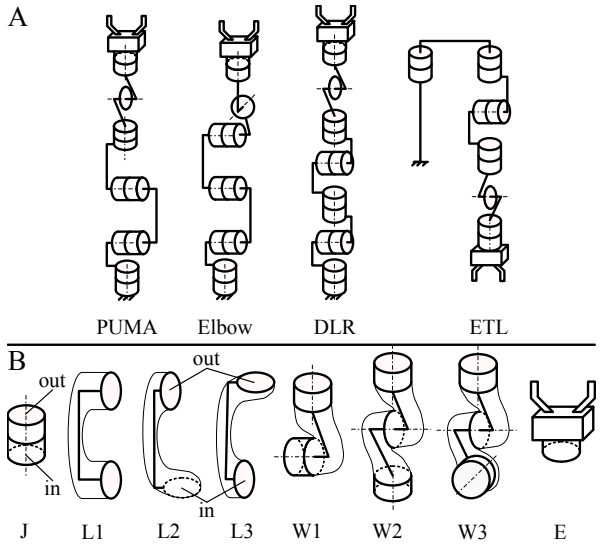

Fig. 2. A: Common kinematics of industrial robots [16], which can be built from B: eight generic module types: Joint (*J*), Link (*L1,L2,L3*), Wrist (*W1,W2,W3*), and Endeffector (*E*).

TABLE III

COMMON INDUSTRIAL ROBOT KINEMATICS WITH ROTARY JOINTS

Robot	**Modules**	# of compositions
PUMA-1	*J–L2–J–L1–J–L3–J–L2–W1–E*	61440
PUMA-2	*J–L2–J–L1–J–L3–W2–E*	8192
Elbow	*J–L2–J–L1–J–L1–W3–E*	8192
DLR	*J–L2–J–L3–J–L2–J–L3–W2–E*	81920
ETL	*J–L1–J–L2–J–L3–W2–E*	8192

A. Enumeration of common industrial robot kinematics

We restrict ourselves to common kinematics of industrial robots with rotary joints, which are the *PUMA*, *Elbow*, and *ETL* kinematics with six degrees-of-freedom (DOF), and the *DLR* kinematics with seven DOF, as shown in Fig. 2A [16], [19]. As Tab. III shows, these kinematics can be composed out of the eight generic module types shown in Fig. 2B. Module type J (joint) has one DOF, module types $W1, W2, W3$ (wrist) have two, three, and three DOF, respectively. In Fig. 3 we show how we categorize the modules of *proModular.1* into these module types. For each robot kinematics, given the total number of modules N_{tot} and the number of variants of each module type N_p, where $p \in [J, L1, ...]$ and the length of the module sequence n the amount of considered compositions reduces drastically from $N = N_{\text{tot}}^n$ to $N = \prod_{j=1}^{n} N_{p_j}$, because $N_{p_j} \ll N_{\text{tot}}$. For our 52 modules, the total number of considered compositions reduces from 52^n to $N = 38486016$, if we only consider common industrial robot kinematics. Furthermore, our modules have two different interface sizes, which should match, when connected to each other, so that the total number further reduces to $N = 167936$.

B. Hierarchical elimination of infeasible compositions

After enumerating the compositions, we eliminate infeasible ones by checking the constraints hierarchically. The presented approach is similar to our previous work [9], but additionally considers full robot dynamics and collisions with complex objects.

The hierarchy order is determined by the amount of computation needed and consists of five steps. To quickly eliminate robots with insufficient size and dynamics we initially perform the following feasibility checks for select poses of the tasks (here: via-points): step 1) the existence of an inverse kinematics solution (3), then step 2) whether the gravity torques computed via the Newton-Euler (NE) Algorithm [19] can be held by the motors (4), and then step 3) whether collisions (5) are present by evaluating (5). In step 4), the previous checks are performed again, but for the complete path by using:

- Case 1: the inverse kinematics trajectory [19],
- Case 2: the interpolated path between all desired poses and the gravity torques via the NE-Algorithm, and
- Case 3: the inverse kinematics solution of the entire workspace.

If, for case 2, there is a collision, then the shortest collision-free path is computed. In this work, we use RRT-connect [17] to generate paths in joint space, while the collision checks are done in Cartesian space. A time-optimal trapezoidal trajectory [18] is then generated following the previously computed path. In step 5) the trajectory is then used to compute the robot dynamics and the NE-Algorithm to check (4) again, while considering external forces. At last, the cost functions of Sec. II can be evaluated using the quantities computed in this section. For case 3, we refer to [20] for efficiently evaluating the dexterity index. The elimination algorithm is also shown in Fig. 4. In the next subsection we detail the generation of models, which are needed for our approach.

C. Automatic robot modelling

Kinematic, dynamic, and geometric modeling of modular robots is required to evaluate the cost functions of Sec. II. For kinematic and dynamic modeling we refer to previous work in [21], while in this subsection, we describe the novel geometric modeling, which is needed for collision checking. We make use of the extended Denavit-Hartenberg (DH) formulation [21], which defines a frame of reference $DH_{\text{ext},i}$ for each link i of the composed robot.

The geometry of a robot link i consists of the the distal part of a joint module J_i (axis i), a link module L, and the proximal part of joint module J_{i+1} (axis $i+1$). The kinematics of each module can be defined using the following 4×4 transformation matrices (see Fig. 5):

- Joint modules: $T_{J_i,1}$ is the transformation between the input frame in and the auxiliary frame a before the joint (see [21]), and $T_{J_i,2}$ between the second auxiliary frame b after the joint and the output frame out.
- Link modules: T_L is the transformation between in and out.

Subsequently, we use superscripts to indicate the reference frame. To define the non-convex geometry of a module, we

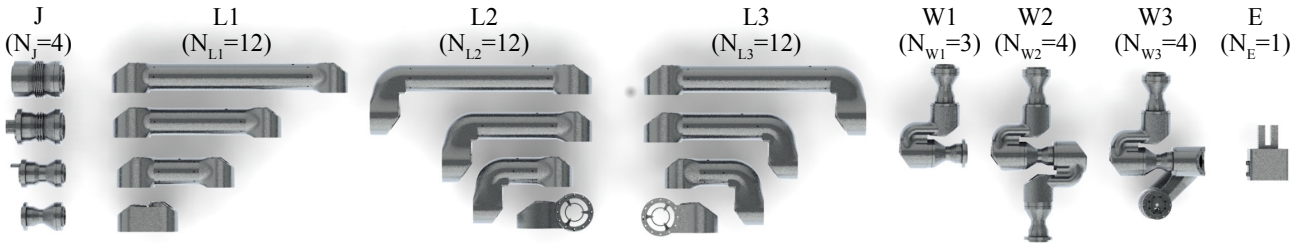

Fig. 3. The *proModular.1* modules are categorized into joints (J), links (L), wrists (W) and end effectors (E) according to Fig. 2B. The number of modules is shown in brackets. Not displayed are module variants with different interface sizes and/or varying motor sizes for wrist modules.

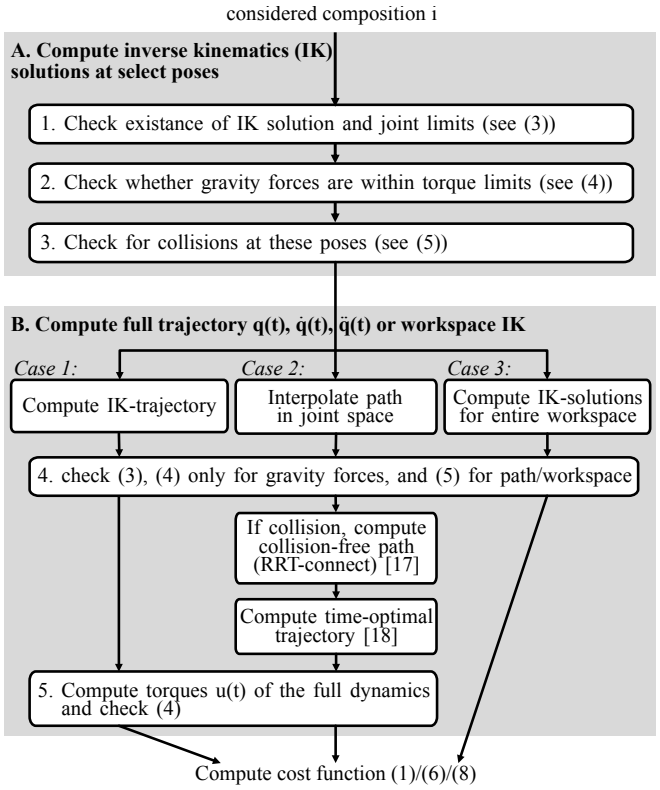

Fig. 4. Hierarchical elimination algorithm with five feasibility checks. The computation for one composition is aborted, if any constraint is not satisfied.

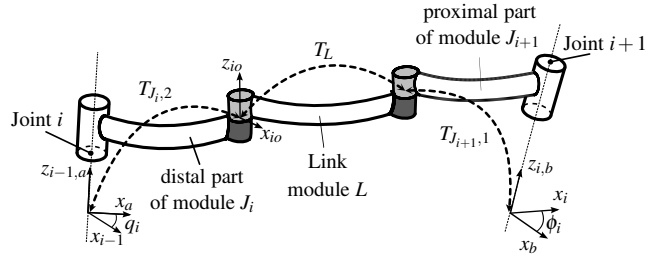

Fig. 5. Illustration of kinematics for synthesizing the geometry of robot link i. Two modules are connected via attaching the output of a module (dark gray) to the input of the next module (light gray), such that the output frame and input frame overlap, e.g., at frame io.

use meshes. A mesh consists of a set of vertices $V \in \mathbb{R}^{3\times m}$ and a set of faces, i.e., triangles defined by the vertices:

- Joint modules: $V_{J,1}$ are vertices relative to auxiliary frame a for the proximal part, and $V_{J,2}$ relative to the output frame for the distal part,
- Link modules: V_L are vertices relative to the output frame.

The geometry of robot link i relative to frame $DH_{\text{ext},i}$ is therefore represented by the set of vertices $\mathcal{V} = \{V^*_{J_i,2}, V^*_L, V^*_{J_{i+1},1}\}$, where

$$\begin{aligned}\begin{bmatrix}V^*_{J_i,2} & 1\end{bmatrix}^T &= R(\phi_i)T^a_{J_{i+1},1}T^{\text{out}}_L \begin{bmatrix}V_{J_i,2} & 1\end{bmatrix}^T\\ \begin{bmatrix}V^*_L & 1\end{bmatrix}^T &= R(\phi_i)T^a_{J_{i+1},1} \begin{bmatrix}V_L & 1\end{bmatrix}^T\\ \begin{bmatrix}V^*_{J_{i+1},1} & 1\end{bmatrix}^T &= R(\phi_i) \begin{bmatrix}V_{J_{i+1},1} & 1\end{bmatrix}^T\end{aligned}$$

where for computing angle ϕ_i we refer to [21]. Standard forward kinematics algorithms can be then used to transform the vertices of each robot link into Cartesian space, where collision checks are performed. Simpler geometries, such as boxes or spheres, can also be used with this approach, but lead to more conservative collision checks.

IV. Evaluation Study

In a simulated evaluation study, we compare the performances of optimized modular robots to commercially available industrial manipulators for randomly generated tasks. Experiment data including all declared parameters, as well as videos showing the randomly generated tasks for this study have been made available to the public [1]. Collision-free paths and collision checks are implemented using the Open Motion Planning (OMPL)[2] and the Fast Collision Checking Library (FCL) [22].

For our comparison, we use a Schunk LWA-4P 6-DOF lightweight robot (*Industrial robot 1*), a KUKA LWR 4+ 7-DOF lightweight robot (*Industrial robot 2*), and a Staubli TX90 robot (*Industrial robot 3*). These robots have been particularly chosen for the availability of their models [23]–[25]. The base coordinate system of all robots is set to the origin $(0, 0, 0)$ without loss of generality, except for ETL-class robots, which are hanging from above, for which we set the base at $(0, 0, 1)$. For opimization, we consider the kinetic and potential of the robots. For case 1 and 2, a random payload between $0...50$ N represents the external force. In

[1] https://git.io/JeYdE
[2] http://ompl.kavrakilab.org

TABLE IV

CASE 1: ENERGY SAVINGS OF MODULAR ROBOTS AND ROBOT MASSES (100 EXPERIMENTS FOR EACH SITUATION)

# poses		Energy savings in % 2	3	4	mass in kg of Ind.R.	Mod.R.
Ind.Rob. 1	μ	31.0	18.4	7.29	13.7	14.7
	Var.	13.6	10.3	14.7		3.97
Ind.Rob. 2	μ	46.2	34.2	28.2	18.7	15.9
	Var.	4.58	4.31	5.12		5.00
Ind.Rob. 3	μ	84.2	83.3	82.7	98.4	17.3
	Var.	0.63	0.36	0.24		7.41

the following, we describe the experiments and their results for each case.

A. Case 1: Desired trajectory

For case 1 we generate trapezoidal trajectories at random constant speeds in Cartesian space connecting, two, three, and four poses for each industrial robot, which results in nine different settings, each with 100 experiments resulting in 900 experiments in total. To make sure these poses can be reached by the respective industrial robot, the poses are randomly generated in their joint spaces instead of the Cartesian space. As a simplification, but without loss of generality, the trajectories do not alter the orientation of the end-effector, and we do not consider external obstacles. The optimal module composition is chosen according to Sec. III. An example of the experiments is seen in Fig. 8 for comparing industrial robot 1 with the optimal modular robot for a trajectory with 4 poses, where the improvement is also displayed.

In Tab. IV we show the mean and variance of the improvement in percent of the energy for each setting. We observe that the energy savings become less when the trajectory is longer. We can see in the comparison with industrial robot 1 that due to clever optimization, energy savings of *proModular.1* are possible despite having a higher robot mass to a large extent due to the customized kinematics. We also see that the energy savings are bigger compared to industrial robot 3, because the average modular robot in these cases have much lower masses.

B. Case 2: Desired poses

Similar to case 1, we automatically sample two, three and four poses in Cartesian space for each industrial robot. Additionally, we consider these scenarios: 1) a scenario without obstacles in the environment, and 2) a pick and place task, where the robot gripper goes in and out of a milling machine. The optimal module composition selects the fastest and energy efficient robots for this task with $[w_e, w_t] = [1, 0.2]$, $\dot{q}_{\max} = 1$ rad/s, and $\ddot{q}_{\max} = 1$ rad/s^2. The results are shown in Tab. V and an example is shown in Fig. 9.

Similar to case 1, time and energy savings are less, when the trajectory length increases. A further decrease in cost savings is visible when considering obstacles. This might be an indication that a higher number of task constraints lead to less room for better performance in a direct comparison. However, what has been neglected in these experiments is that through optimization and adaptation, a modular robot is limitless when it comes to performing tasks, that would be infeasible for fixed-shape industrial robots.

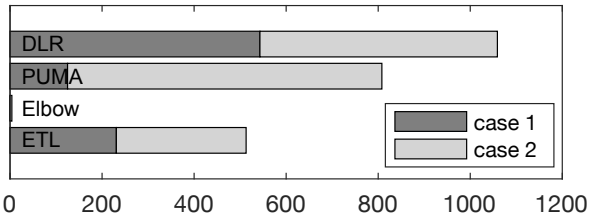

Fig. 6. Case 1 and 2: Appearance frequency of each kinematics as a result of the optimization.

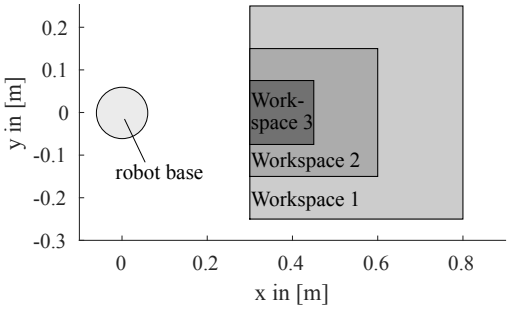

Fig. 7. Case 3: Three desired cubical workspaces, for which we optimize the dexterity. The lower bounds of the workspaces are all at $z = 0$. The robot base is placed at $[0, 0, 0]$.

Another interesting outcome is shown in Fig. 6, which presents the frequency a kinematics has been chosen in all experiments of case 1 and 2. The DLR kinematics has been chosen by far the most for case 1, which supports the fact that kinematic redundancy leaves more room for trajectory optimization in inverse kinematics control. For case 2, the PUMA kinematics is chosen the most. Elbow robots have almost never been selected.

C. Case 3: Desired workspace

In this experiment we showcase the results of optimizing the dexterity index of workspaces. The three investigated workspaces are shown in Fig. 7. We consider poses that lead to self-collisions as not reachable. The results for the optimal modular robot, as well as the three industrial robots are shown in Tab. VI. It displays clearly that a dexterity > 0.7 can always be achieved (maximum is 1), while such is not the case for industrial robots.

V. CONCLUSIONS & DISCUSSION

In this work we show a computationally feasible approach for the task-based optimization of modular robots based on the enumeration of common industrial robot kinematics and on hierarchical elimination, that involves feasibility checks of the kinematics, dynamics, and obstacle collision checking of the composed robot. By limiting the enumeration to standard (but practically proven) kinematics, we reduce computational load, but also omit non-standard designs that may perform even better. In some respects, our approach can be viewed

TABLE V

CASE 2: ENERGY AND TIME SAVINGS OF MODULAR ROBOTS – WITH AND WITHOUT OBSTACLES (100 EXPERIMENTS PER SITUATION)

		no obstacles						with obstacle (machine tooling example)					
		Time savings in %			Energy savings in %			Time savings in %			Energy savings in %		
number of poses		2	3	4	2	3	4	2	3	4	2	3	4
Industrial robot 1	μ	33.1	22.2	10.0	46.4	26.3	6.52	27.8	17.3	8.70	38.9	4.61	−5.18
	Var.	4.13	3.04	3.06	10.6	9.57	12.8	4.12	6.00	4.33	11.9	9.87	18.5
Industrial robot 2	μ	37.0	28.3	21.8	69.0	63.8	51.7	37.2	32.7	18.3	63.1	59.6	47.4
	Var.	3.90	2.59	1.89	2.06	2.64	2.61	1.95	3.79	3.23	2.77	5.25	3.42
Industrial robot 3	μ	43.4	30.1	21.6	88.0	83.5	80.9	27.4	30.4	46.3	86.1	81.8	89.7
	Var.	4.80	1.89	3.22	1.09	1.09	0.46	7.65	0.82	0.57	0.45	0.27	0.03

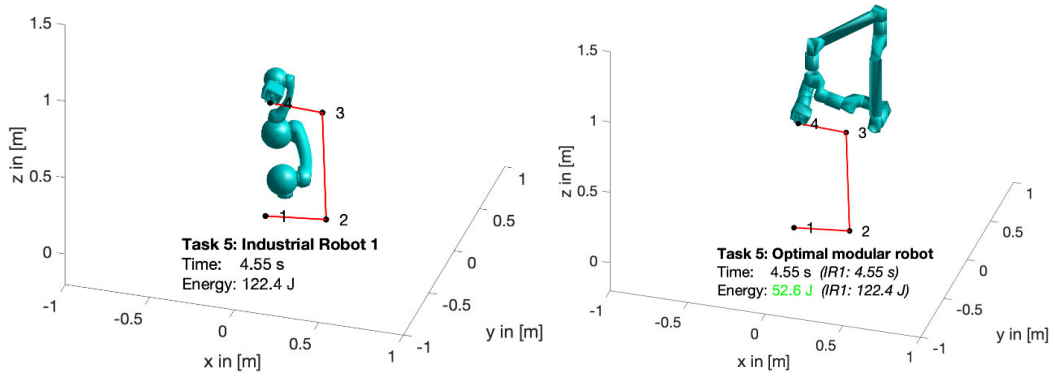

Fig. 8. Case 1: exemplary comparison of industrial robot 1 with the optimal module composition (ETL) for a trajectory with four points

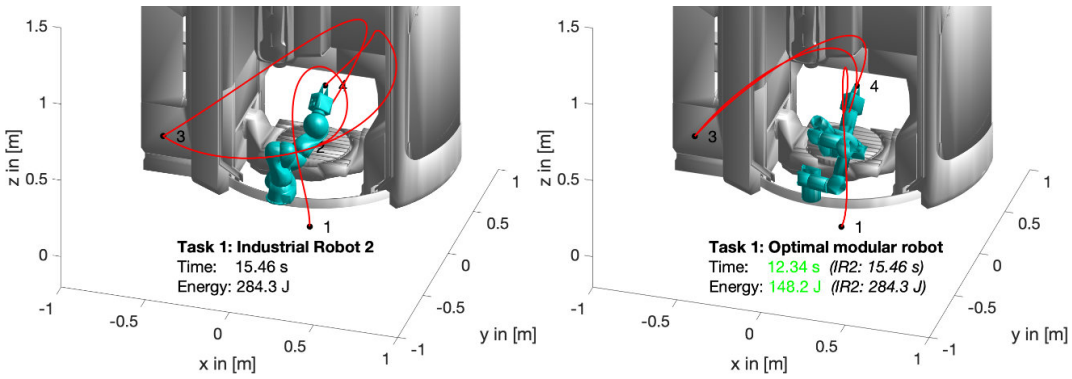

Fig. 9. Case 2: exemplary comparison of industrial robot 2 with the optimal module composition (PUMA) for a list of four desired poses and an obstacle

TABLE VI

CASE 3: OPTIMIZED WORKSPACE DEXTERITY INDICES COMPARED WITH INDUSTRIAL ROBOTS

Work-space	Optimal Mod.rob.	Industrial robot 1	2	3
1	0.859	0.496	0.298	0.323
2	0.782	0.723	0.254	0.354
3	0.730	0.794	0.186	0.363

as a discretized parameter optimization, where the kinematic structures are given. However, one can manually add more kinematics in the future (e.g., SCARA with translational joints), which then only linearly increases the number of enumerated compositions. Interestingly, it is also possible to use our approach to evaluate the usefulness of kinematic structures, as shown in Fig. 6.

In an evaluation we show that our *proModular.1* robot often achieves better performance for standard tasks in manufacturing than three commercially available industrial robots. In the future, we also aim at evaluating the repeatability performance of modular robots, as well as experiments on the real robots. We believe that this work provides a path towards broader deployment of modular robots in industrial automation.

ACKNOWLEDGMENT

The authors gratefully acknowledge financial support by the Central Innovation Programme of the German Federal Government under grant ZF4086004LP7. We especially thank Paul Maroldt, Hanna Krasowski, Florian Skiba, and Daniel Gheorgita for their contributions in designing *proModular.1* and Constantin Dresel for implementing the collision avoidance planning.

REFERENCES

[1] M. Althoff, A. Giusti, S. B. Liu, and A. Pereira, "Effortless creation of safe robots from modules through self-programming and self-verification," *Science Robotics*, vol. 4, no. 31, p. eaaw1924, 2019.

[2] M. Wilson, *Implementation of Robot Systems*. Elsevier, 2015.

[3] Paryanto, M. Brossog, M. Bornschlegl, and J. Franke, "Reducing the energy consumption of industrial robots in manufacturing systems," *International Journal of Advanced Manufacturing Technology*, vol. 78, no. 5-8, pp. 1315–1328, 2015.

[4] S. Patel and T. Sobh, "Manipulator Performance Measures - A Comprehensive Literature Survey," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 77, no. 3-4, pp. 547–570, 2014.

[5] C. J. Paredis and P. K. Khosla, "Synthesis Methodology for Task Based Reconfiguration of Modular Manipulator Systems," in *Proc. of International Symposium on Robotics Research*, 1993, pp. 2–5.

[6] R. J. Alattas, S. Patel, and T. M. Sobh, "Task-Based Design of Modular Robots: Evolutionary Approach," in *Proc. of IEEE International Conference on Mechatronics and Automation*, 2018, pp. 1787–1793.

[7] I.-M. Chen and J. W. Burdick, "Enumerating the Non-Isomorphic Assembly Configurations of Modular Robotic Systems," *International Journal of Robotics Research*, vol. 17, no. 7, pp. 702–719, 1998.

[8] S. Singh, A. Singla, and E. Singla, "Modular Manipulators for Cluttered Environments: A Task-Based Configuration Design Approach," *Journal of Mechanisms and Robotics*, vol. 10, no. 5, p. 051010, 2018.

[9] E. Icer, A. Giusti, and M. Althoff, "A task-driven algorithm for configuration synthesis of modular robots," in *Proc. of IEEE International Conference on Robotics and Automation*, 2016, pp. 5203–5209.

[10] S. Farritor, S. Dubowsky, N. Rutman, and J. Cole, "A systems-level modular design approach to field robotics," in *Proc. of IEEE International Conference on Robotics and Automation*, vol. 4, 1996, pp. 2890–2895.

[11] O. Chocron and P. Bidaud, "Genetic design of 3D modular manipulators," in *Proc. of IEEE International Conference on Robotics and Automation*, vol. 1, 1997, pp. 223–228.

[12] G. Yang and I. M. Chen, "Task-based optimization of modular robot configurations: minimized degree-of-freedom approach," *Mechanism and Machine Theory*, vol. 35, no. 4, pp. 517–540, 2000.

[13] Q. Li and J. Zhao, "A universal approach for configuration synthesis of reconfigurable robots based on fault tolerant indices," *Industrial Robot: An International Journal*, vol. 39, no. 1, pp. 69–78, 2012.

[14] M. Brandstoetter, "Adaptable Serial Manipulators in Modular Design," Ph.D. dissertation, UMIT, 2016.

[15] E. Icer, H. A. Hassan, K. El-Ayat, and M. Althoff, "Evolutionary cost-optimal composition synthesis of modular robots considering a given task," in *Proc. of IEEE/RSJ International Conference on ts and Systems*, 2017, pp. 3562–3568.

[16] J. Lloyd and V. Hayward, "Kinematics of common industrial robots," *Robotics and Autonomous Systems*, vol. 4, no. 2, pp. 169–191, 1988.

[17] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. of IEEE International Conference on Robotics and Automation*, vol. 2, 2004, pp. 995–1001.

[18] T. Kunz and M. Stilman, "Time-Optimal Trajectory Generation for Path Following with Bounded Acceleration and Velocity," *Robotics: Science and Systems VIII*, pp. 209–216, 2012.

[19] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics. Modelling, Planning and Control*. London: Springer, 2009.

[20] O. Porges, R. Lampariello, J. Artigas, A. Wedler, C. Borst, and A. Roa, "Reachability and Dexterity: Analysis and Applications for Space Robotics," in *Proc. of the Workshop on Advanced Space Technologies for Robotics and Automation*, 2015.

[21] A. Giusti and M. Althoff, "Automatic centralized controller design for modular and reconfigurable robot manipulators," *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3268–3275, 2015.

[22] J. Pan, S. Chitta, and D. Manocha, "FCL: A general purpose library for collision and proximity queries," *Proc. of IEEE International Conference on Robotics and Automation*, pp. 3859–3866, 2012.

[23] S. B. Liu and M. Althoff, "Reachset Conformance of Forward Dynamic Models for the Formal Analysis of Robots," in *Prof. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 370–376.

[24] A. Jubien, M. Gautier, and A. Janot, "Dynamic identification of the Kuka LWR robot using motor torques and joint torque sensors data," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 8391–8396, 2014.

[25] J. Jin and N. Gans, "Parameter identification for industrial robots with a fast and robust trajectory design approach," *Robotics and Computer-Integrated Manufacturing*, vol. 31, no. 1, pp. 21–29, 2015.