A Reconfiguration Strategy for Modular Robots using Origami Folding

The International Journal of Robotics Research
XX(X):1–16

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Meibao Yao[1], Christoph H. Belke[2], Hutao Cui[1] and Jamie Paik[2]

Abstract

Reconfigurability in versatile systems of modular robots is achieved by changing the morphology of the overall structure as well as by connecting and disconnecting modules. Recurrent connectivity changes can cause misalignment that leads to mechanical failure of the system. This paper presents a new approach to reconfiguration, inspired by the art of origami, that eliminates connectivity changes during transformation. Our method consists of an energy-optimal reconfiguration planner that generates an initial 2D assembly pattern and an actuation sequence of the modular units, both resulting in minimum energy consumption. The algorithmic framework includes two approaches, an automatic modeling algorithm as well as a heuristic algorithm. We further demonstrate the effectiveness of our method by applying the algorithms to Mori, a modular origami robot, in simulation. Our results show that the heuristic algorithm yields reconfiguration schemes with high quality, compared to the automatic modeling algorithm, simultaneously saving a considerable amount of computational time and effort.

Keywords

Modular robotics, Optimal reconfiguration, Computational origami, Heuristic algorithm

1 Introduction

Modular reconfigurable robots are versatile systems capable of transforming their shape to perform various tasks in different environments. In mission-based scenarios such as space applications Zykov et al. (2007); Yao et al. (2018c), recognition and sampling Guzman et al. (2016) or search and rescue Zhang et al. (2006); Wang et al. (2014), where handling the complexity and uncertainty of environments is beyond the capabilities of conventional fixed-morphology robots, modular robotic systems can autonomously reconfigure to adapt to new circumstances. However, due to the large number of individual entities and degrees-of-freedom (DoFs) in modular systems, the planning and execution of changing a configuration is highly challenging.

Several approaches to planning and optimizing this process have been proposed thus far, which can be grouped into two main categories. The first category considers a target configuration by dividing the modular architecture into different sets of modules in order to simplify the reconfiguration process. A task-oriented optimization has been implemented by minimizing the number of DoFs of different types of modules in the configuration Yang and Chen (2000). A multi-objective optimization problem has been addressed with a concurrent design approach that splits the architecture into different levels Bi and Zhang (2001), a neural network approach that finds the optimal design Fei et al. (1998), as well as a genetic algorithm Wu et al. (2016). The relationship between morphological design and motion planning has been established and analyzed by using the implicit function theorem to optimize design parameters Ha et al. (2017). The aforementioned approaches facilitate the optimal design of the final configuration for specific tasks, but do not address the dynamic reconfiguration procedure in modular robots.

The second category of optimizing the reconfiguration process aims at minimizing the number of connectivity changes when transforming from an initial shape into the desired configuration. The challenge of such a combinatorial optimization problem lies in its inherent NP-completeness Casal and Yim (1999), even for chain-type modular robots Hou and Shen (2010), yielding the need for heuristic solutions with some guarantee of performance. A graph-based planning algorithm has thus been proposed to generate a near-optimal transforming sequence in polynomial time Hou and Shen (2014).

Although some of the above reconfiguration planners target a reduction in the number of connectivity changes, all of the approaches require some form of disconnect-and-connect between modules in the process. Connectivity changes complicate the overall transformation, are time-consuming and can lead to failure resulting from commonly experienced alignment errors between modules Kurokawa et al. (2008);

[1]Harbin Institute of Technology, China
[2]École Polytechnique Fédérale de Lausanne (EPFL), Switzerland

Corresponding author:
Jamie Paik, Reconfigurable Robotics Lab (RRL), École Polytechnique Fédérale de Lausanne (EPFL), Switzerland.
Email: jamie.paik@epfl.ch

Meibao Yao is with the Deep Space Exploration Research Center, Harbin Institute of Technology, China, email:meibaoyao@gmail.com

Wei et al. (2010). Consequently they cast high demands on control algorithms and hardware implementations to detect and overcome misalignments Romanishin et al. (2013). It is thus desirable to minimize or even remove any connectivity changes from a reconfiguration process.

In this work we introduce a planning algorithm for the reconfiguration of modular robots that does not require any connectivity changes during the transformation, but only before motion begins and after it is completed. For this purpose we make use of the fact that in task-specific scenarios modular robots commonly transform from a collection of modules on a surface to a functional three-dimensional configuration. We can therefore simplify the reconfiguration process by turning a two-dimensional (2D) pattern into a 3D shape, given that a corresponding 2D pattern exists Wang et al. (2016); Yao et al. (2018b). This process, in its conceptual form, can be related to the principles behind origami, the art of folding sheets of paper into 3D objects Paik et al. (2012); Firouzeh and Paik (2015); Zhakypov et al. (2017); Yao et al. (2018a). Furthermore, by also considering the inverse of this process we can find a way of transforming from one 3D configuration to another with an intermediate 2D phase during which all connectivity changes occur, greatly reducing the possibility of misalignment errors.

This paper introduces a novel, optimized method for planning the reconfiguration of modular robotic systems by drawing similarities to the concept of folding in origami. While utilizing origami principles to plan the reconfiguration process eliminates the need for connectivity changes during transformation, a remaining critical aspect for modular systems is energy consumption. Since most systems have an on-board power-supply, minimizing power consumption is vital, particularly during the highly-demanding reconfiguration process. The overall reconfiguration planning problem can thus be formulated as follows: Given a desired 3D target configuration, find the 2D layout and the corresponding actuation sequence that will result in the lowest energy consumption.

The energy consumption during reconfiguration depends on the initial planar pattern as well as the actuation sequence of different modules, both of which are essential considerations in the field of computational origami Peraza-Hernandez et al. (2014), committed to designing algorithms that compute the folding process of origami structures. Computing the initial crease pattern of a sheet of paper, corresponding to an initial planar pattern of modular robots, has been optimized for paper models using tree-based algorithms Lang (1996), while the rigid foldability of objects, where only flat sections can exist between hinges, is subject to mathematical conditions Watanabe and Kawaguchi (2009). The rigid origami model applies directly to our approach since it omits the properties of paper creases, allowing direct kinematic translations between tiles or modules Tachi (2009, 2010); Filipov et al. (2015). While the aforementioned approaches assume zero-thickness structures, additional constraints need to be applied when the folding object has a non-zero thickness Chen et al. (2015); Zirbel et al. (2013). Automating the folding process of both zero and non-zero thickness structures requires planning the sequence of folding steps. This has been demonstrated by means of a robot capable of folding origami Balkcom and Mason (2008) as well as self-folding robotic sheets An and Rus (2014). Previous approaches to modeling algorithms for folding origami structures consider a wide range of problems and scenarios; however, they do not consider energy consumption during the reconfiguration process.

This paper presents a new approach to reconfiguration for modular robotic systems inspired by origami principles, as well as the first algorithmic framework for planning an energy-optimal reconfiguration scheme of folding in modular robots. In our approach we determine an optimal initial planar configuration and an optimal actuation sequence, NP-complete problems where the candidates of reconfiguration schemes grow exponentially with the number of modules in the robotic system. Since the optimization scheme cannot be validated without enumerating the search space, we develop an automatic modeling algorithm for modular robots based on Hollerbach (1980); Chen and Yang (1998); Chen et al. (1999); Meister et al. (2013), as well as a heuristic algorithm for generating energy-optimal folding schemes. The proposed methodology is validated by demonstration on a modular robotic platform, Mori, a modular origami robot Belke and Paik (2017). The main contributions of this paper are as follows:

1. An automatic modeling algorithm to generate the kinematic model and dynamic derivation of robotic aggregates, represented by the hierarchical structure of a rooted tree. The torque consumption of pre-folding patterns is calculated for pre-defined folding sequences with motion planning that considers the thickness of the structure.
2. A heuristic algorithm for energy-optimal folding of modular robots. This includes an optimal 2D layout planner followed by two folding sequence planners, one unified actuation planner for different layouts and one optimal planner within a specific layout, proving the validity of algorithms.
3. The proposed algorithms are demonstrated on several 3D configurations of the Mori platform. The performance of the energy-optimal reconfiguration schemes are evaluated by comparing the automatic modeling algorithm and the heuristic algorithm.

2 Methodology

In this section we present the first algorithmic framework for planning an energy-optimal reconfiguration scheme of modular robots using origami principles. The method overview is illustrated in Fig. 1, and the algorithmic framework is composed of two main approaches, *automatic modeling algorithm* and *heuristic algorithm* respectively. A target 3D shape determined by mission requirements is given as input to the algorithms. In general there are large quantities of possible folding schemes incorporating pre-folding patterns and actuation orders of modules, that can finally shape the target 3D configuration. Due to the NP-completeness of the optimal reconfiguration planning problem, enumeration procedures are necessary to search for the optimal solution. An intuitive and direct approach is to calculate the consumed torque of different folding schemes,

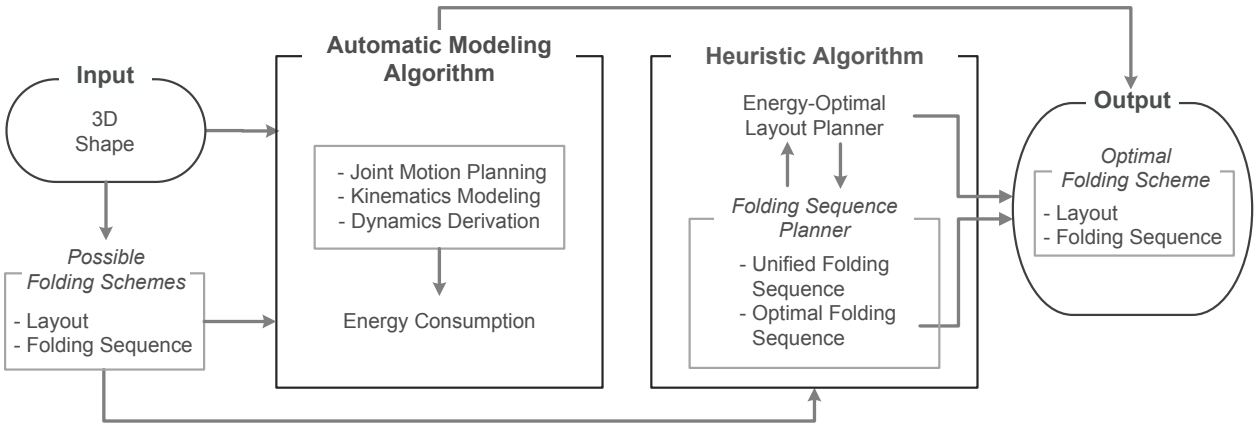


Figure 1. Overview of energy-optimal reconfiguration planning for modular robots using two approaches, an automatic modelling algorithm and a heuristic algorithm, as presented and applied in this work.

and we propose an *automatic modeling algorithm* (Sec. 3) to follow through this approach. In this algorithm, joint motion is first planned for each module in the pre-folding pattern, considering folded state constraints due to thickness of modules. Forward kinematics of the modular architecture is then automatically generated with its geometry modeled as a hierarchical structure of rooted tree. Joint dynamics of the robotic configuration is then derived with recursive Newton-Euler (RNE) formulation, and energy consumption of the reconfiguration is calculated with the predefined folding sequence. A less straightforward approach is to design a *heuristic algorithm* (Sec. 4) to evaluate torque consumption of different folding schemes, by utilizing some measurement of energy expense for assessment based on the derived joint dynamics. The proposed algorithm includes two heuristic planning procedures. The first is a two-step planning process to figure out the pre-folding pattern with minimum energy consumption, by first predefining a unified folding sequence (Sec. 4.2.2) aimed at eliminating the coupling effect of actuation order and layout in consumed torque, followed by an optimal layout planner (Sec. 4.1) for execution. The second procedure determines the folding sequence with optimality for the energy-optimal layout generated in the first process, by implementing an optimal folding sequence planner (Sec. 4.2.3). The output of the two algorithmic approaches is energy-optimal reconfiguration schemes of optimized layout folding with ideal actuation order of modules.

3 Automatic Modeling Algorithm

The modeling of modular robot systems is a difficult issue where the extensive approaches for fixed-structure robots cannot be directly applied, because the number of robotic configurations grow exponentially with the number of modules Liu et al. (2010). For this reason, automated modeling techniques Chen and Yang (1998); Chen et al. (1999); Meister et al. (2013) were proposed and developed to deal with the modeling complexity. The dynamic model of modular robots is generally derived by RNE method. Most of the approaches are developed based on Lie theory Park and Bobrow (1994), where adjoint mapping is conducted on the Lie group $SE(3)$ of homogeneous transformations. The adjoint representation in the derived wrench and torque is a compact expression of matrix, which complicates the derivation and estimation for energy consumption of joint actuators, usually in an integration form with time Wu et al. (2016); Kamimura et al. (2005). In this section we develop an *automatic modeling algorithm* for modular robots following Hollerbach (1980), aimed at calculating energy consumption of folding schemes during the reconfiguration. The overall procedure of the *automatic modeling algorithm* is shown in Alg. 1, and the algorithmic inputs and step-by-step modeling techniques of the algorithm are explained in detail as follows.

Algorithm 1 Automatic modeling algorithm.

Input: A 3D shape; a folding scheme (a pre-folding pattern and a folding sequence);
Output: Energy consumption of the folding scheme;
1: Joint motion planning with joint angle constraints;
2: Automatic kinematics modeling of the modular architecture;
3: Autonomous dynamics derivation of joint motion;
4: Energy consumption computed with folding sequence;
5: **return** Torque consumption of reconfiguration.

3.1 Input

The input of the *automatic modeling algorithm* in Alg. 1 are the mesh Straub and Prautzsch (2011) of a target 3D shape and a folding scheme, including a pre-folding pattern and a folding sequence. In general, for a 3D configuration, there are numerous planar patterns that can finally fold up into it. The reciprocal process of reconfiguration, unfolding a 3D structure into 2D layouts Straub and Prautzsch (2011), is utilized to generate pre-folding patterns. The general procedure is to produce graph representation of the 3D configuration, followed by spanning trees and corresponding rooted trees generated by the represented graph. A rooted tree stands for a pre-folding pattern of the 3D structure with a designated root, and can be generated via traversal algorithms, such as Breadth First Search (BFS) and Depth First Search (DFS). The connected structure of a rooted tree remains unchanged during the reconfiguration as no connectivity rearrangement is required in folding using origami principles. The rooted tree thus represents the

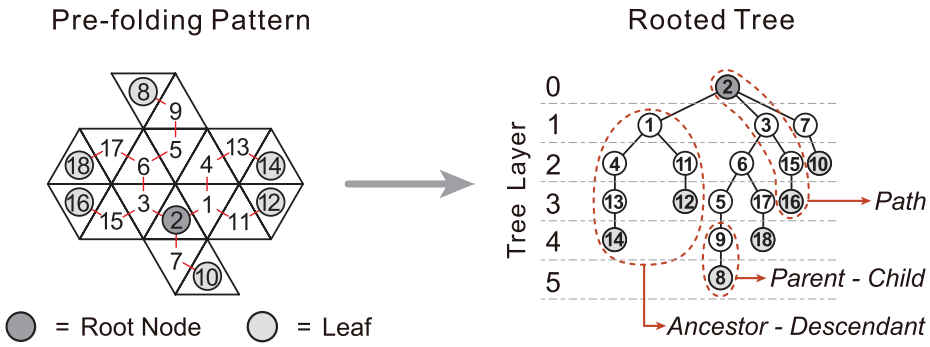


Figure 2. A pre-folding pattern with a root node and its corresponding rooted tree. Left: A planar layout with a designated root representing the robotic base of folding motion. Right: The rooted tree creates hierarchical structure of the layout. Important components in the architecture include *leaf*, *path*, and *parent - child*, *ancestor - descendant* relationships. The tree layer describes distance between a vertice and the root node along its *path*.

intermediate quasi-2D shape and desired 3D configuration simultaneously. A pre-folding pattern with root node and its corresponding rooted tree are shown in Fig. 2. Regarding the geometry of a pre-folding pattern, each modular unit can be cubic-shaped or hexagonal plate, or even irregular shapes with additional DoFs in a heterogeneous robotic system. Here we assume regular-shaped units of triangles in a homogeneous system that remains general to any lattice shape. The root node is the ancestor of all other modules and shapes unique tree structure in kinematics modeling, as well as the origin of inertial frame in dynamics derivation and torque computation. Different root nodes selected as the robotic base, result in variations of the required torque, therefore it needs to be specified. The hierarchical structure of the modular architecture represented by a rooted tree is utilized for its geometric characterization in both kinematics modeling and dynamics derivation. Another input is the folding sequence of modules, which correlates the time intervals of the reconfiguration, and will be discussed in Sec. 4.2.

3.2 Joint Motion Planning

The first step of the algorithm is to plan joint trajectories for modular units in time periods consistent with the folding sequence. It is obvious that torque consumption varies with different joint motion profiles, even with the same folding scheme. Therefore, a unified joint motion planning for every module in each time period is a guarantee of fair comparison of torque consumption for different folding schemes. With the discussion on time periods and folding sequence detailed in Sec. 4.2, we focus on the joint trajectory planning in each time period as follows.

To generate a joint trajectory passing through several path control points, predefined knots on the trajectory, a series of polynomials are usually used to interpolate the path Guan et al. (2005). The joint angle $q_i(t)$ of module i varies from zero initial condition with the planar pattern to its final position $\hat{\theta}_i$ in time period $T_k \in [t_{k-1}, t_k]$, $k \in \mathbb{Z}^+$, as $q_i(t_{k-1}) = 0$ and $q_i(t_k) = \hat{\theta}_i$, with default zero boundary conditions for velocity and acceleration, as $\dot{q}_i(t_{k-1}) = \dot{q}_i(t_k) = 0$ and $\ddot{q}_i(t_{k-1}) = \ddot{q}_i(t_k) = 0$. The folding angle $\hat{\theta}_i$ is the dihedral angle between module i and its *parent* in the 3D structure, and can be computed by the algorithm presented in An et al. (2014). Since there are 6 boundary conditions mentioned above as path control points, a 5th order polynomial is needed for the interpolation. We note that the range of joint angle is constrained if thickness of the module is considered in joint motion planning, as depicted in Fig. 3. The threshold value, denoted as ϑ, of joint angle $q_i(t)$ due to thickness of the module can be calculated as

$$\sin(\frac{\vartheta}{2}) = \frac{d/2}{r} \tag{1}$$

where d is the thickness of the modular entity and r is the radius of the hinge on each lateral of the module, and the joint angle is thus constrained as

$$\begin{cases} -\pi < q_i(t) < -\vartheta \\ \vartheta < q_i(t) < \pi \end{cases} \tag{2}$$

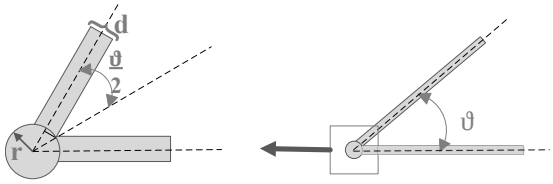


Figure 3. The minimum joint angle between two modules, ϑ, is constrained by the thickness of a module.

3.3 Kinematics Modeling

The next step of the algorithm (Alg. 1 Step 2) is to generate a kinematic model of the robotic configuration and automatically calculate folding motion of each module, including position, velocity and acceleration, given a set of joint trajectories yielded in step 1. This procedure is widely known as forward kinematics Meister et al. (2013) in modular robot modeling. In a modular architecture modeled as a hierarchical structure defined by a rooted tree as Fig. 2, the kinematic transformation $\mathbf{\Phi}$ starting from root node R to each of the *descendant* modules i in *path* j can be recursively formulated as $\mathbf{\Phi}_i = \boldsymbol{A}_{\boldsymbol{P}_{j,1}}^{\boldsymbol{P}_{j,2}} \cdot \boldsymbol{A}_{\boldsymbol{P}_{j,2}}^{\boldsymbol{P}_{j,3}} \cdots \boldsymbol{A}_{\boldsymbol{P}_{j,s-1}}^{\boldsymbol{P}_{j,s}}$, where $\boldsymbol{A}_k^{k'}$ is a 4×4 homogeneous matrix describing the relative pose of neighboring modules (k, k'), and can be calculated according to Meister et al. (2013). $\boldsymbol{P}_j$ is the set of modules in jth *path* of the rooted tree, and $\boldsymbol{P}_{j,1} = R$, $\boldsymbol{P}_{j,s} = i$.

3.4 Dynamics Derivation

In Step 3 of Alg. 1, dynamics of the robotic configuration is derived autonomously via modified RNE formulation, and input torque of joint motors can be calculated for each folding scheme. Here we assume zero ground interactions for simplification of the mathematical derivation, which is common practice in related literature Chen and Yang (1998); Chen et al. (1999); Meister et al. (2013). In the application to modular robots, the initialization and backward recursion are similar to standard RNE Hollerbach (1980). In backward recursion, where generalized velocity and acceleration of each module are propagated from root node to *leaves* (the end modules) of all *paths*, the robotic motion does not interfere with multiple connections in a module, due to the uniqueness of the *parent* module of each unit in the rooted tree structure. However, when forces and moments are propagated from the *leaves* to the root node R in forward recursion, *children* modules interfere with them. Therefore, the forward recursion propagating forces and moments exerted on module i by its *parent* can be derived as:

$$\begin{aligned}
\boldsymbol{f}_i &= \sum_{j=1}^{|\boldsymbol{C}_i|} \boldsymbol{f}_{\boldsymbol{C}_{i,j}} + \boldsymbol{F}_i \\
\boldsymbol{n}_i &= \sum_{j=1}^{|\boldsymbol{C}_i|} (\boldsymbol{n}_{\boldsymbol{C}_{i,j}} + \boldsymbol{p}_i \times \boldsymbol{f}_{\boldsymbol{C}_{i,j}}) + \boldsymbol{N}_i + (\boldsymbol{p}_i + \boldsymbol{r}_i^*) \times \boldsymbol{F}_i \\
\tau_i &= \boldsymbol{z}_i \cdot \boldsymbol{n}_i
\end{aligned} \tag{3}$$

with

$$\begin{aligned}
\boldsymbol{F}_i &= m_i \ddot{\boldsymbol{r}}_i \\
\boldsymbol{N}_i &= \boldsymbol{I}_i \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times (\boldsymbol{I}_i \boldsymbol{\omega}_i)
\end{aligned} \tag{4}$$

where related variables and parameters are listed in Tab. 1.

In this formulation the reference coordinate frame is the base coordinates, an implicit expression compared to more commonly used link coordinates, which is considered to be more efficient. The formulation with base coordinates, however, reveals the direct relations between the required torque and link motion, including position and orientation in the inertial frame. This is advantageous for evaluation and estimation of energy consumption for the *heuristic algorithm* design (Sec. 4).

3.5 Energy Consumption

The final procedure (Alg. 1 Step 4) is to evaluate energy consumption of a reconfiguration scheme, with modules folding in series. This depends on the type of joint motors; for a commonly used DC motor, for example, the consumed energy is expressed as the product of required torque and angular velocity of the joint Wu et al. (2016). For computational simplification, here we assume that the total energy consumed by all motors is an accumulated value of torque during the time span of evaluation. Another assumption is that locking mechanism is incorporated into the modular unit design, constraining joint motion of unactuated modules as commanded, ensuring no input torque required to maintain the configurations of modular aggregates. The energy consumption of the folding scheme, denoted as ε, with n modules in the configuration, during global folding time $t \in [t_0, t_p]$ can be evaluated by the following equation:

$$\varepsilon = \sum_{i=1}^{n} \frac{\int_{t_{k-1}}^{t_k} |\tau_i| \, dt}{t_k - t_{k-1}} = \sum_{i=1}^{n} \frac{\int_{t_0}^{t_p} |\tau_i| \, dt}{t_k - t_{k-1}} \tag{5}$$

With the above *automatic modeling algorithm* (Alg. 1), energy consumption of the input folding scheme, with a specific layout and folding order of modules, is calculated automatically. Enumerate torque consumption of possible folding schemes with the evaluation function in Eq. 5, and the energy-optimal reconfiguration scheme can be specified.

Table 1. Variables and parameters in Sections 3.4 and 4.1

	Variable
$\boldsymbol{f}_i$	Force exerted on module i by its *parent*;
$\boldsymbol{n}_i$	Moment exerted on module i by its *parent*;
τ_i	Input torque of actuator along $\boldsymbol{z}_i$;
$\boldsymbol{z}_i$	Actuating axis of module i, defined as the coincident edge of module i and its *parent*;
$\boldsymbol{F}_i$	Total external force on module i;
$\boldsymbol{N}_i$	Total external moment on module i;
$\boldsymbol{p}_i$	Vector from DH coordinate origin $i-1$ to coordinate origin i;
$\boldsymbol{r}_i$	Vector from the base coordinate origin to CM of module i;
$\ddot{\boldsymbol{r}}_i$	Acceleration vector of $\boldsymbol{r}_i$;
$\boldsymbol{r}_i^*$	Vector from DH coordinate origin i to the center of mass (CM) of module i;
$\hat{r}_i$	Distance between robotic base (root node R) and CM of module i, and $\hat{r}_i(t) = \lvert\boldsymbol{r}_i(t)\rvert$;
$\boldsymbol{\omega}_i$	Angular velocity of module i;
$\dot{\boldsymbol{\omega}}_i$	Angular acceleration of module i;
$\boldsymbol{C}_i$	The set of *child/ children* of module i and $\lvert\boldsymbol{C}_i\rvert$ is the number of components in $\boldsymbol{C}_i$.
	Module parameter
l	Side length of each modular unit;
m_i	Mass of module i;
$\boldsymbol{I}_i$	Inertia tensor of module i about its CM.
	Other parameter
λ_s	$s = 1, 2, 3, 4$, λ_s are constants defined as $\lambda_1 = \lVert\boldsymbol{I}_i\rVert$, $\lambda_2 = \frac{l/2}{\sin(\pi/3)} m_i$, $\lambda_3 = m_i$, and $\lambda_4 = \lvert\boldsymbol{p}_i\rvert = 0$ or l;
κ_s	$s = 1, 2, 3$, $\kappa_s > 0$ are design parameters.

4 Heuristic Algorithm

Calculating torque consumption by modeling reconfiguration procedures is not the only method to generate the energy-optimal folding scheme, if some measurement or estimation of energy expense can be utilized to assess the reconfiguration schemes. In contrast to the direct but computation-heavy strategy with the *automatic modeling algorithm* (Sec. 3), we propose a *heuristic algorithm* to evaluate torque consumption of a folding scheme. This includes two planning procedures, aimed at determining an optimal pre-folding pattern and a folding sequence of modules with minimum consumed energy.

4.1 Energy-optimal Layout Planner

Among numerous planar patterns generated by the unfolding procedures (Sec. 3.1), it is desirable to pre-assemble the optimal layout that will consume minimum torque when transforming its shape into a target 3D structure, before execution of reconfiguration. It is therefore important to specify such an energy-optimal pre-folding pattern in the planning problem. Inspired by creating optimal cut-out sheets in a 3D papercraft model Straub and Prautzsch (2011); Haenselmann and Effelsberg (2012), where weight functions are put forward with some optimum criteria and assigned to edges of a represented graph, here we design a cost function that evaluates torque consumption of a pre-folding pattern during its reconfiguration.

In a rooted tree structure as depicted in Fig. 2, the edge between *parent* and *child* can be assigned a weighted value measuring energy consumption of actuating the module of *child* to its final position. The weight function of edge $e_i : (i, i')$, where i' is the *parent* of module i, is denoted as $\varpi(e_i)$. The overall torque consumption for a pre-folding pattern can be evaluated by a cost function χ, the sum of weights with all edges in the rooted tree.

$$\chi = \sum_{i=1}^{n-1} \varpi(e_i) \tag{6}$$

The measurement of energy consumption in actuation time $T_k \in [t_{k-1}, t_k]$ of module i can be implemented by integrating $|\tau_i|$ twice with time, denoted as $\varepsilon_i' = \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\tau_i| \, dtdt$, and the energy metric can be derived from joint dynamics in Eq. 3 as follows.

$$\varepsilon_i' = \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} ||\boldsymbol{z}_i| \cdot |\boldsymbol{n}_i| \cdot cos(\varphi_i)| \, dtdt \tag{7}$$

where φ_i is the vector angle between $\boldsymbol{z}_i$ and $\boldsymbol{n}_i$.

It is noted that $\boldsymbol{z_i} = \boldsymbol{\Phi}_i' \boldsymbol{z}_0$, where $\boldsymbol{\Phi}_i'$ is a orthonormal rotation matrix calculated as the upper 3×3 of the kinematic transform matrix $\boldsymbol{\Phi}_i$. $\boldsymbol{z}_0 = [0\ 0\ 1]$ is a unit vector in z direction of the inertial frame, thus $|\boldsymbol{z}_i| = 1$. With $cos(\varphi_i) \leq 1$, ε_i' in Eq. 7 can be evaluated as

$$\varepsilon_i' \leq \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{n}_i| \, dtdt \tag{8}$$

Define $\Lambda_i = \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{n}_i| \, dtdt$ and $\Gamma_i = \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{f}_i| \, dtdt$, and Theorem 1 is stated as follows.

Theorem 1. *The energy metric of actuating module i to its final position $\hat{\theta}_i$, denoted as ε_i', can be estimated by Λ_i^*, defined as the maximum of Λ_i, and Λ_i^* is derived from Eq. 3 as*

$$\begin{aligned} \Lambda_i^* = &\sum_{j=1}^{|\boldsymbol{C}_i|} (\Lambda_{\boldsymbol{C}_{i,j}}^* + \lambda_4 \cdot \Gamma_{\boldsymbol{C}_{i,j}}^*) + \lambda_1 \cdot (\kappa_2 \left|\dot{\hat{\theta}}_i\right| + \kappa_3 \left|\dot{\hat{\theta}}_i\right|^2) \\ &+ \lambda_2 \kappa_1 \left|\hat{r}_i(t_k) - \hat{r}_i(t_{k-1})\right| \end{aligned} \tag{9}$$

with Γ_i^*, the maximum of Γ_i written as

$$\Gamma_i^* = \sum_{j=1}^{|\boldsymbol{C}_i|} \Gamma_{\boldsymbol{C}_{i,j}}^* + \lambda_3 \kappa_1 \left|\hat{r}_i(t_k) - \hat{r}_i(t_{k-1})\right| \tag{10}$$

where related variables and parameters are listed in Tab. 1.

Proof. See Appendix.

The calculation procedures of Λ_i^* and Γ_i^* are consistent with that of joint torque τ_i in Eq. 3, which is a forward recursion propagating from *leaf* to module i along the *path*. The computation requires updated position data of the robotic configuration with the term $\hat{r}_i(t)$, determined by the actuation order of modules in time periods, detailed in the folding sequence planner design (Sec. 4.2). This procedure is inherently a kinematic propagation rather than the integration of joint torque with time as in Eq. 7. Hence, the simplified estimation by Eq. 9 substantially reduces the computational cost. The parameter κ_s is designed according to joint motion profile, including interpolation method and boundary conditions. With unified joint trajectory predefined for modules presented in Section 3, κ_s is set to be identical with each modular unit, as well as for computational simplification.

The energy metric, defined as twice integration of torque magnitude has equivalence with the energy consumption defined as Eq. 5 for comparing and ranking energy consumption among reconfiguration schemes. This metric allows us to derive its upper bound, defined above as Λ_i^*, and calculate it by variables such as joint angle and inertial distance, instead of their derivatives and second derivatives. In the computational process, this can significantly reduce computing load and simplify the optimization procedures. Therefore, the weight function $\varpi(e_i)$ can be designed according to Λ_i^*, as $\varpi(e_i) = \Lambda_i^*$, to estimate the energy metric of module i in its movement. The evaluation of torque consumption in a pre-folding pattern can be implemented by the cost function in Eq. 6. Enumerate possible layouts of the target 3D shape with this estimation procedure, and obtain the one with minimum cost, which is the energy-optimal layout generated by the heuristic planner.

4.2 Folding Sequence Planner

In reconfiguration planning for modular robots, the aggregate of modules in a pre-folding pattern can be actuated simultaneously to perform folding motion, which is an obvious choice for timesaving reconfiguration in an urgent task. However, a synchronous actuation scheme without considering folding sequence does not guarantee energy-optimal folding for a layout. Hence, actuation time and energy consumption are two coupled and contradictory objectives in reconfiguration planning. Challenges arise when trading off both objectives and determining that some modules move while others remain stationary in series, emerging as a NP-complete problem. In this section, we propose two algorithms of asynchronous folding sequence planners, including a unified framework for generating actuation order in different layouts, and an energy-optimal folding sequence planner with timesaving consideration in a specific layout respectively.

4.2.1 Problem Formulation

For a pre-folding pattern with n modules, there are $n-1$ modules except the one of root node R, to be actuated with p folding steps in series. Each folding step spans from time t_{k-1} to t_k, as time period $T_k \in [t_{k-1}, t_k]$, $k = 1, 2, \cdots, p$. Assume that the

length of time periods is equal and denoted as Δt, then $t_k = t_{k-1} + \Delta t$ and $t_p = t_0 + p\Delta t$. There are n_k $(n_k > 0)$ modules in T_k to be actuated to their final folding angles and others stay still, and it is obvious that $\sum_{k=1}^{p} n_k = n - 1$. To achieve timesaving reconfiguration simultaneously, the number of modules folding up in each time period should be as large as possible. The goal of a folding sequence planner is to determine which n_k $(n_k > 0)$ modules of a layout to be folded up in time period T_k, with minimum energy consumption and sequence number p. In this planning we assume no collisions occur among modules during the reconfiguration.

4.2.2 Unified Folding Sequence To compare torque consumption of different layouts during reconfiguration, a set of unified rules should be established to predefine joint motion of each module (Sec. 3.2), as well as the folding sequence for each layout. For the reason that different actuation orders of modules can lead to variation in energy consumption even for the same pre-folding pattern, folding sequence generated under a uniform framework ensures a consistent comparison. Here we present a unified folding sequence planner to decouple the effect of folding sequence from the pre-folding pattern in energy consumption.

In most cases, torque consumption of folding module i to its final angle, varies if it is actuated in different time periods, since movement of a module interferes in the structure with constraints between connected modules. It is noted that only when modules are in the same *path*, their movement may change position with each other, resulting in distinct torque consumption with a different order of folding. To decouple the interference of movement in energy consumption, it is supposed that:

Assumption 1. Modules in the same *path* of the rooted tree of a layout are actuated in separate time periods.

It is obvious that the simplest connection with other modular entities occurs when the module is *leaf* in a rooted tree, thus modules of *leaves* in the pre-folding pattern are first investigated. If module i is *leaf*, then $|\boldsymbol{C}_i| = 0$, from Eq. 3, joint torque of module i is simplified as:

$$\begin{aligned}\tau_i &= \boldsymbol{z}_i \cdot [\boldsymbol{N}_i + (\boldsymbol{p}_i + \boldsymbol{r}_i^*) \times \boldsymbol{F}_i] \\ &= \boldsymbol{z}_i \cdot [\boldsymbol{I}_i \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times (\boldsymbol{I}_i \boldsymbol{\omega}_i) + (\boldsymbol{p}_i + \boldsymbol{r}_i^*) \times m_i \ddot{\boldsymbol{r}}_i]\end{aligned} \tag{11}$$

which shows that torque τ_i is only determined by the movement and position of module i, also validated by the definition that *leaf* has no connection with other modules except its *parent*. Since *leaf* of a rooted tree $\boldsymbol{\Upsilon}$, denoted as $leaf(\boldsymbol{\Upsilon})$, are in different *paths*, it is straightforward that all *leaves* can be actuated in the first time period T_1, for the purpose of timesaving reconfiguration with minimum number of folding steps p. With the locking mechanism, *leaf* is attached to its *parent* after the actuation. Previous *leaf/ leaves* being folded up could be eliminated from $\boldsymbol{\Upsilon}$ and a new rooted tree is obtained. In the next time period T_2, all new *leaves* of $\boldsymbol{\Upsilon}$ are modules to be actuated. Similarly, in time period T_k, the set of modules to be folded up, denoted as $\boldsymbol{\Omega}_k$, are new *leaf/ leaves* of $\boldsymbol{\Upsilon}$, and $|\boldsymbol{\Omega}_k| = n_k$. This process is carried on until only one node (root node R) is left in the rooted tree $\boldsymbol{\Upsilon}$. The algorithmic procedure to uniformly predefine folding sequence for different layouts is summarized in Alg. 2. In the actual implementation, the unified folding sequence planner can be utilized to automatically generate actuation orders for different layouts in an enumeration process with the energy-optimal layout planner (Sec. 4.1) or the *automatic modeling algorithm* (Sec. 3).

Algorithm 2 Unified folding sequence planner.

Input: Rooted tree $\boldsymbol{\Upsilon}$ of a pre-folding pattern;
Output: Sets of modules folding in series;
1: Initialization: $k \leftarrow 1$, $n \leftarrow |\boldsymbol{\Upsilon}|$, $\boldsymbol{\Omega}_k \leftarrow \emptyset$;
2: **while** $n > 1$ **do**
3: $\quad \boldsymbol{\Omega}_k \leftarrow leaf(\boldsymbol{\Upsilon})$, *leaves* of $\boldsymbol{\Upsilon}$ are modules to be actuated in time period T_k;
4: $\quad \boldsymbol{\Upsilon} \leftarrow \boldsymbol{\Upsilon} - \boldsymbol{\Omega}_k$, cut the edge of $leaf(\boldsymbol{\Upsilon})$ and their *parents*, and obtain a new rooted tree;
5: $\quad n \leftarrow |\boldsymbol{\Upsilon}|$
6: $\quad k = k + 1$
7: **end while**
8: **return** $\boldsymbol{\Omega}_1, \boldsymbol{\Omega}_2, \cdots, \boldsymbol{\Omega}_k, \cdots, \boldsymbol{\Omega}_p$

4.2.3 Optimal Folding Sequence Folding from an initial 2D layout, to find out the energy-optimal actuation order of modules is complex because a folding sequence cannot be verified as optimal without evaluating torque consumption with all possible folding orders within the layout. To calculate the exact energy consumption with automatic modeling technique (Section. 3) is not only computationally laborious but time-consuming. In this section, a heuristic planner is proposed to estimate the energy consumption of a folding sequence scheme.

The unified folding sequence planner (Sec. 4.2.2) takes advantage of simple connection of *leaf* in a rooted tree, but for other modules that are not *leaf*, torque consumption of the movement couples with the configuration and position of their *descendant*, as can be inferred from Eq. 3. With the locking mechanism, the motion of modules can be described by rigid body rotation around the actuating axis under gravity. Therefore, joint torque of actuating motor of module i is simplified as:

$$\tau_i = J_i \ddot{q}_i - M_{g,i} \tag{12a}$$

$$J_i \ddot{q}_i = \sum_{j=1}^{\left|\boldsymbol{D}_i^{'}\right|} I_{j,i}^{'} \ddot{q}_i \tag{12b}$$

$$M_{g,i} = \left| \boldsymbol{r}_{g,i} \times \frac{\boldsymbol{z}_i \times \left(\sum_{j=1}^{\left|\boldsymbol{D}_i^{'}\right|} m_j \boldsymbol{g} \times \boldsymbol{z}_i \right)}{|\boldsymbol{z}_i|^2} \right| \tag{12c}$$

where

$\boldsymbol{D}_i^{'} = \{\boldsymbol{D}_i, i\}$ is set of modules, with $\boldsymbol{D}_i$ denotes set of *descendant* of module i;

J_i moment of inertia of $\boldsymbol{D}_i^{'}$ with respect to $\boldsymbol{z}_i$ (fixed during movement of module i);

$M_{g,i}$ gravity torque of $\boldsymbol{D}_i^{'}$ relative to $\boldsymbol{z}_i$;

$I_{j,i}^{'}$ moment of inertia of module j with respect to $\boldsymbol{z}_i$;

$\boldsymbol{g} = \begin{bmatrix} 0 & 0 & -9.8 \end{bmatrix} m^2/s$ is gravitational acceleration vector;

$\boldsymbol{r}_{g,i}$ perpendicular vector from $\boldsymbol{z}_i$ to CM of $\boldsymbol{D}_i^{'}$.

The work-energy principle is a general principle that can be applied specifically to rotating objects. For pure rotation, the net work is equal to the change in rotational kinetic energy. The net work of module i is denoted as W_i and according to the work-energy principle, it is derived as

$$W_i = W_{\tau,i} + W_{g,i} = \frac{1}{2} J_i \dot{q}_i(t_k)^2 - \frac{1}{2} J_i \dot{q}_i(t_{k-1})^2 \quad (13)$$

with $W_{\tau,i}$, work of τ_i during time period T_k

$$W_{\tau,i} = \int_{q(t_{k-1})}^{q(t_k)} \tau_i(t)\, dq \quad (14)$$

and $W_{g,i}$, work of gravity torque during T_k

$$W_{g,i} = (\sum_{j=1}^{|\boldsymbol{D}_i^{'}|} m_j)(\boldsymbol{r}_{g,i}(t_k) \cdot \boldsymbol{g} - \boldsymbol{r}_{g,i}(t_{k-1}) \cdot \boldsymbol{g}) \quad (15)$$

With the predefined joint motion profile (Sec. 3.2), at the initial and terminal moment of each time period T_k, the joint velocity of each module is under zero boundary conditions, as $\dot{q}_i(t_{k-1}) = \dot{q}_i(t_k) = 0$. It is thus obvious that $W_i = 0$, and $W_{\tau,i} = -W_{g,i}$, according to Eq.13. Energy measurement of joint motor of module i can be represented by $W_{\tau,i}$, and energy consumption of folding motion with certain folding sequence W_τ is written as:

$$W_\tau = \sum_{i=1}^{n} |W_{\tau,i}| \quad (16)$$

Therefore, an evaluation function χ' can be designed to assess energy consumption of a folding sequence scheme, represented by W_τ, according to Eqs. 15 and 16.

$$\begin{aligned} \chi' = W_\tau &= \sum_{i=1}^{n} |W_{g,i}| \\ &= \sum_{i=1}^{n} \left| (\sum_{j=1}^{|\boldsymbol{D}_i^{'}|} m_j)(\boldsymbol{r}_{g,i}(t_k) \cdot \boldsymbol{g} - \boldsymbol{r}_{g,i}(t_{k-1}) \cdot \boldsymbol{g}) \right| \end{aligned} \quad (17)$$

With the above function χ', an enumeration process can be executed to figure out the energy-optimal folding sequence for a layout.

The asynchronous actuation of modular units can be executed one after the other, but this process can be rearranged into several folding steps for timesaving reconfiguration purpose under Assumption 1. Since the motion of modules in different *paths* with module i would not influence joint torque of the module, torque consumption with various folding sequences can be identical. It indicates that modules in different *paths* can be actuated in the same time period T_k without changing energy consumption of the scheme. Although permutations of folding order of n modules in a layout are as many as $n!$, the number of schemes of folding sequence can be reduced significantly. Given an array of module sequence $\boldsymbol{S} = [s_1, s_2, \cdots, s_i, \cdots, s_j, \cdots, s_n]$, $s_i, s_j \in \{1, 2, \cdots, n\}$, $s_i \neq s_j$, it can be rearranged into p folding steps instead of n. The rearranged folding sequence is denoted as $\{\boldsymbol{\Omega}_1, \boldsymbol{\Omega}_2, \cdots, \boldsymbol{\Omega}_p\}$, and $\boldsymbol{P}_{\boldsymbol{i}}^{'}$ represents the set of modules in the same *path* with module i. The algorithmic procedure of rearranging a vector of folding order into folding steps is presented in Alg. 3.

Algorithm 3 Rearrangement of folding sequence.

Input: $\boldsymbol{S} = [s_1, s_2, \cdots, s_n]$;
Output: $\boldsymbol{\Omega}_1, \boldsymbol{\Omega}_2, \cdots, \boldsymbol{\Omega}_k, \cdots, \boldsymbol{\Omega}_p$;
Initialization: $k \leftarrow 1$, $n \leftarrow |\boldsymbol{S}|$, $\boldsymbol{\Omega}_k \leftarrow \emptyset$;
while $n > 1$ **do**
  $\boldsymbol{\Omega}_k \leftarrow \boldsymbol{\Omega}_k \cup \{s_1\}$
  for $i = 2 \rightarrow n$ **do**
    $\boldsymbol{P}_{\boldsymbol{i}}^{'} \leftarrow \emptyset$
    for $j = 1 \rightarrow |\boldsymbol{\Omega}_k|$ **do**
      $a \leftarrow \boldsymbol{\Omega}_k(j)$
      $\boldsymbol{P}_{\boldsymbol{i}}^{'} \leftarrow \boldsymbol{P}_{\boldsymbol{i}}^{'} \cup \{a\}$
    end for
    if $s_i \notin \boldsymbol{P}_{\boldsymbol{i}}^{'}$ **then**
      $\boldsymbol{\Omega}_k \leftarrow \boldsymbol{\Omega}_k \cup \{s_i\}$
    end if
  end for
  $k \leftarrow k + 1$
  $\boldsymbol{S} \leftarrow \boldsymbol{S} - \boldsymbol{\Omega}_k$
  $n \leftarrow |\boldsymbol{S}|$
end while
return $\boldsymbol{\Omega}_1, \boldsymbol{\Omega}_2, \cdots, \boldsymbol{\Omega}_k, \cdots, \boldsymbol{\Omega}_p$

The number of steps p of folding sequence generated by Alg. 2 and 3 is equal to the *height* of rooted tree of the pre-folding pattern, defined as length of the longest path, for the reason that in each time period, at least one module in a *path* will be actuated. p is thereby the minimum number of actuation steps in the sense of Assumption 1, and timesaving reconfiguration is achieved through the unified folding sequence planner (Alg. 2) and the algorithm for rearrangement of folding sequence (Alg. 3).

5 Simulation and Experiment

To evaluate performance of the proposed methods and algorithms using origami principles in this paper, we utilize Mori Belke and Paik (2017), a modular robotic system with origami features, as an experimental platform. The system overview, as well as the reconfiguration and unfolding procedures of Mori will be discussed in this section. The energy-optimal reconfiguration schemes are generated by both *automatic modeling algorithm* and *heuristic algorithm*, and comparative results of simulation are provided.

5.1 System Overview

The platform of Mori is a homogeneous robotic system, and each modular unit is triangular-shaped and fabricated using a multi-jet 3D printer. There are two types of modules in the system, active and passive. The active module is integrated with electronic components, with three stepper motors embedded on each lateral for actuation. The module properties of the Mori platform are summarized in Table. 2. To generate the kinematic model of robotic configuration in the *automatic modeling algorithm*, Denavit-Hartenberg

(a) Pre-folding pattern **(b)** Folding motion **(c)** 3D configuration

Figure 4. The reconfiguration procedure of the Mori robotic platform. (a). A planar pre-folding pattern is shaped when each triangular module connects with other modular units. (b). The aggregates are actuated with a controlled sequence of modules to perform folding motion during the reconfiguration. (c). The desired 3D configuration of a quadruped comes into conformation.

Table 2. Module properties and DH parameters of Mori

Module properties Belke and Paik (2017)		
l	Side length $[mm]$	80
d	Thickness $[mm]$	6
m	Mass $[g]$	26 (Active module) 8 (Passive module)
I	Inertia tensor $[g.mm^2]$	$10^{-2} \times [-3.09, -3.09, -6.15, 0.67, 8.32 \times 10^{-2}, 2.30 \times 10^{-16}]$
DH parameters		
ζ_1	Offset along previous $\boldsymbol{z}$ to common normal of previous $\boldsymbol{z}$ and new $\boldsymbol{z}$	$\zeta_1 = 0$ *or* l
$\zeta_2(q)$	Angle about previous $\boldsymbol{z}$, from old $\boldsymbol{x}$ to new $\boldsymbol{x}$	Joint variable
ζ_3	Length of the common normal	$\zeta_3 = 0$ (Rotation axis intersect)
ζ_4	Angle about common normal, from old $\boldsymbol{z}$ to new $\boldsymbol{z}$	$\zeta_4 = 60°$

(DH) parameters of Mori architecture with triangular-shaped modules are also defined in Tab. 2.

5.2 Reconfiguration and Unfolding

Combining the characteristics of reconfigurable modular robots and foldable origami-inspired robotic systems, Mori allows the execution of complex tasks in various environments. The reciprocal procedures of reconfiguration and unfolding with Mori platform are discussed as follows.

5.2.1 Reconfiguration of Mori As well as folding into a static and complicated structure as origami, Mori can transform its shape into dynamic configurations according to task requirements, such as locomotion and transportation of components. The reconfiguration procedure of Mori is illustrated in Fig. 4, and when in the 3D conformation of quadruped, it can conduct dynamic movements in a task-oriented scenario.

5.2.2 Unfolding 3D Configurations The demonstration of proposed methodology is realized in two mission-based scenarios, including statically functional structures e.g., supporting truss, infrastructure, and complicated tasks requiring dynamic movements. Three target 3D shapes, a Tetrahedron, an Octahedron and a locomotion configuration of Quadruped in Fig. 4c are utilized in the simulation. The desired 3D structures of Mori and their possible pre-folding patterns are shown in Tab. 3. Through unfolding process (Section. 3.1) via graph representation, all possible layouts can be generated and some of them are depicted. We assume the same 2D pattern with various arrangements of modules as different layouts, such as Tetrahedron L1 and Tetrahedron L13.

5.3 Energy-optimal Folding Schemes

Torque consumption varies with different reconfiguration schemes involving root node, pre-folding patterns and folding sequence. Therefore, the folding scheme of Mori is composed of a 2D layout folding with a designated root node as robot base and a predetermined folding sequence of modules. In this section. we present the simulation results for energy-optimal reconfiguration schemes, including optimal layouts with unified folding sequence and an optimal folding sequence planner designed for pre-folding patterns.

5.3.1 Optimal Layout Optimal layouts of a 3D configuration presented herein is in the sense of predetermining folding sequence by a unified folding sequence planner (Alg. 2), with a root node R chosen as robot base. Parameters in Theorem 1 of the heuristic optimal layout planner (Sec. 4.1) are summarized in Tab. 5. Results of evaluating energy-optimal layouts generated by two methods are illustrated in Tab. 4, and visualization of layouts in Tab. 4 and corresponding unified folding sequence are presented in Tab. 6. Method I (MI) is to calculate torque consumption of

Table 3. Target 3D configurations and corresponding pre-folding patterns

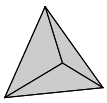
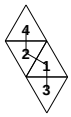
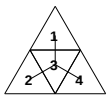
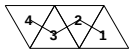
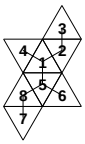
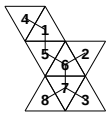
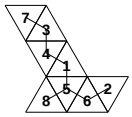
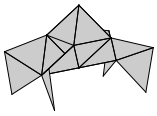
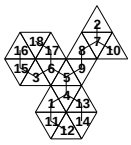
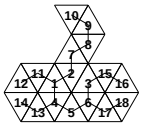
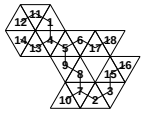
3D shape	Possible layouts (L for Layout)		
			
Tetrahedron	Tetrahedron L1	Tetrahedron L7	Tetrahedron L13
			
Octahedron	Octahedron L1	Octahedron L160	Octahedron L320
			
Quadruped	Quadruped L35	Quadruped L70	Quadruped L105

Table 4. Evaluation of energy-optimal layouts

3D shape	Method	Optimal layout	Torque consumption $[Nm]$	Maximum torque consumption $[Nm]$	Ranking from MI
Tetrahedron	MI	L3 $(R=1)$	0.0156	0.3210	—
	MII	L3 $(R=1)$	0.0156		$1/16$
Octahedron	MI	L331 $(R=5)$	0.2400	2.3279	—
	MII	L6 $(R=5)$	0.2944		$14/384$
Quadruped	MI	L1 $(R=5)$	0.9895	4.8916	—
	MII	L1 $(R=5)$	0.9895		$1/108$

all possible layouts with the *automatic modeling algorithm* (Alg. 1), and delivers the energy-optimal one, as presented in Section 3. Method II (MII) utilizes the proposed heuristic energy-optimal layout planner (Sec. 4.1) to estimate torque consumption of layouts. The ranking of MII from MI is explained as the optimal layout generated by MII ranked out of possible layouts sorted by torque consumption, calculating from MI in ascending order. For the 3D shapes of Tetrahedron and Quadruped, optimal layouts produced by two methods with chosen root node R are identical. For Octahedron, the layout generated by Method II ranks 14 out of 384 candidates that can fold into a Octahedron.

Table 5. Parameters in Theorem 1

Constant parameter		Design parameter	
λ_1	6.15×10^{-11}	κ_1	3
λ_2	0.0012	κ_2	1
λ_3	0.026	κ_3	2
λ_4	0 or 0.08		

A more detailed comparison of the two methods generating energy-optimal layouts was implemented on a Windows PC with Intel Core i7, 2.6 GHz, 12G RAM, and the results are shown in Tab. 7. With each root node R chosen as robotic base among n modules, as $R=1,2,\cdots,n$, energy-optimal layouts are generated by two methods. Assume that there are n_1 optimal layouts generated by MII identical with those yielded by MI, and n_2 optimal layouts generated by MII, with less torque consumption than the median of all layouts with the chosen root node R. The performance of MII is measured by two rates: Rate 1 is calculated as $\frac{n_1}{n}\times100\%$, indicating equivalence in validity of two methods and optimality of MII; Rate 2 is computed by $\frac{n_2}{n}\times100\%$, suggesting the superiority of layouts generated by MII compared to randomly selected pre-folding patterns. It is noted that computation time of MII is significantly less than MI, which can be explained by the simplification of estimating torque consumption with MII, instead of computing the integral of joint torque with MI. With increase in complexity of 3D shapes, measured by module and layout quantities, the performance of MII deteriorates, but Rate 2 is still larger than 50% in the worst case. The results in Tab. 7 indicate that the proposed heuristic energy-optimal layout planner carried out by MII can be applied to real-time planning of energy-optimal reconfiguration according to mission requirements, with desired guarantee that the generated layout consumes less energy than randomly chosen ones.

5.3.2 Optimal Folding Sequence Optimal folding order of a pre-folding pattern can be sorted out by enumeration with Eq. 17, and the vector of folding order of modules is rearranged to a folding sequence with several steps via Alg. 3. Optimal folding sequence determined for optimal layouts generated by MII in Tab. 4 are shown in Fig. 5. Since there

Table 6. Energy-optimal layouts with unified folding sequence

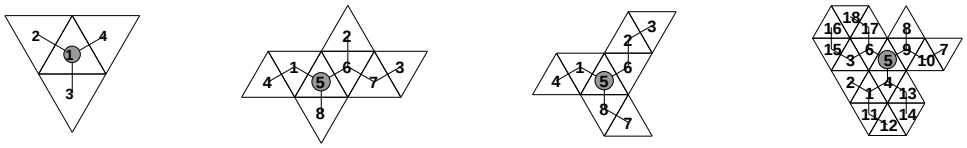
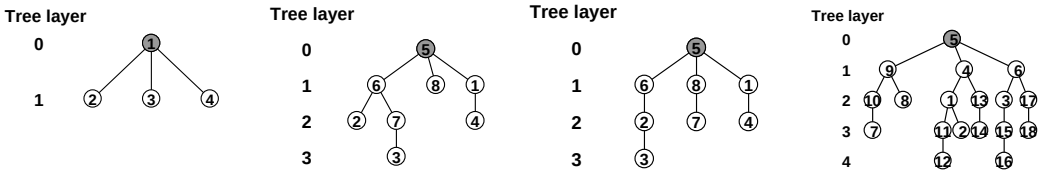
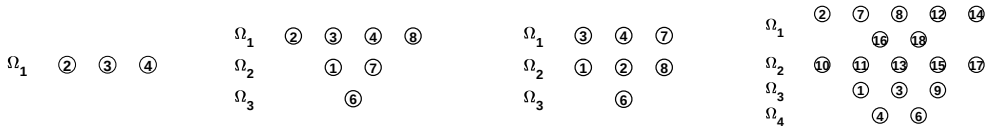
	Tetrahedron L3 $(R=1)$	Octahedron L331 $(R=5)$	Octahedron L6 $(R=5)$	Quadruped L1 $(R=5)$
Optimal layout				
Rooted tree				
Folding sequence				

Table 7. Comparison of methods generating energy-optimal layouts

3D shape	Module quantity	Layout quantity	Method	Computation time $[s]$		Performance	
				Total	Per layout	Rate 1	Rate 2
Tetrahedron	4	16	MI	36	0.5625	—	—
			MII	0.72	0.0113	100%	100%
Octahedron	8	384	MI	6435	2.09	—	—
			MII	12.50	0.0041	0	87.50%
Quadruped	18	108	MI	13723	7.06	—	—
			MII	20	0.0103	5.56%	61.11%

is only one folding sequence of Tetrahedron L3 $(R=1)$ in Tab. 6, we provide the visualized results of reconfiguration schemes with optimality for Octahedron L6 $(R=5)$ and Quadruped L1 $(R=5)$.

The order of magnitude of possible folding orders is at least $\left(\lceil \frac{1}{3}n \rceil\right)!$ for the Mori system, which would still be large for enumeration if the module number n grows, thus Monte-Carlo method is utilized to verify the optimality of folding sequence of 3D shapes in Fig. 5. Arrays of folding order of a pre-folding pattern are generated randomly and rearranged as folding sequence with Alg. 3. Torque consumption of folding layouts with optimal folding sequence and randomly generated ones are calculated via Alg. 1. Different layouts of various 3D shapes are utilized in the Monte-Carlo verification of the optimal folding sequence planner (Sec. 4.2.3), as illustrated in Fig. 6. Simulation results show that the evaluating rate is larger than 70% in the worst case of Quadruped L70 $(R=5)$, and the performance of heuristic planner is 100% better than randomly generated actuation orders in three selected layouts out of eight. This indicates that the proposed folding sequence planner generates actuation order of modules with small torque consumption, compared to large numbers of folding schemes. The Monte-Carlo simulation in Fig. 6 validates the effectiveness of the proposed heuristic solution of optimal folding sequence planner.

6 Discussion

The focus of the presented work is to minimize connection misalignments in modular robots by pre-assembling the initial configuration and sequentially actuating modular units without docking and locking during the morphological transformation. This approach of origami folding reconfiguration is not only feasible for a modular robotic system with low-profile modules such as Mori, but is also possible for a large number of general modular robots with homogeneous or heterogeneous modules. Certain conditions should be satisfied for such feasibility. Firstly, systematic design of each module should have at least one DoF of rotation, which most modular robots do. This condition ensures that quasi folding motion is allowed in the robotic system and can be carried out by a sequence of actuation. Secondly, initial configuration space should include planar patterns that can finally transform into the target 3D shape, guaranteeing that the set of feasible solutions of the reconfiguration planning problem is nonempty. Therefore, this new methodology for generating energy-optimal folding schemes can be applied to a wide range of modular robotic systems.

To go a step further, for more general reconfiguration of modular robots, that is, starting from an arbitrary 3D shape to form another 3D shape, our approach still applies in energy-optimal shape-to-shape transition. The initial 3D shape should be first unfolded into a 2D pattern, which can be optimized by utilizing the proposed reconfiguration planner

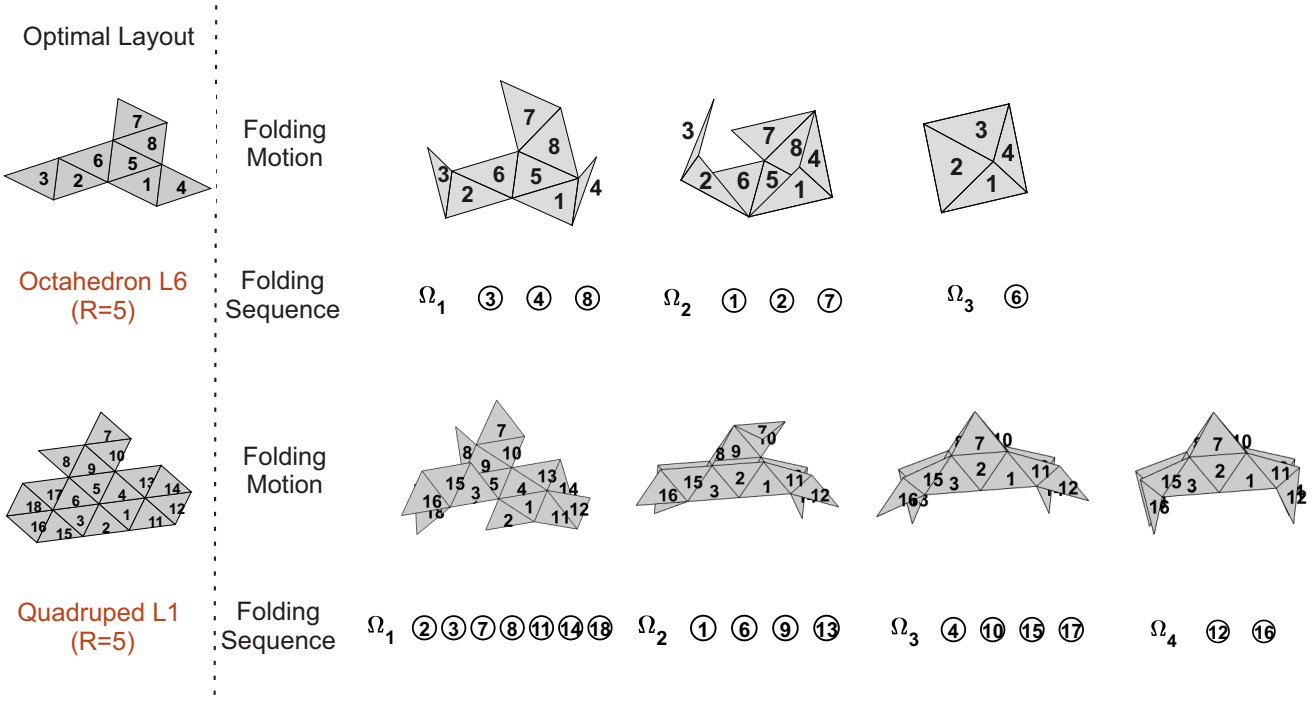


Figure 5. Reconfiguration sequences for optimal folding schemes of layouts L6 and L1 for an octahedron and a quadruped, respectively, as determined by the optimal folding sequence planner (Sec. 4.2.3).

for an energy-optimal unfolding scheme. The generated 2D pattern can be assembled to form an energy-optimal initial 2D pattern of the target 3D shape, and finally folded into the target 3D configuration.

Regarding the assumption in the algorithms, we suppose that locking mechanism is included in modular entities. It prevents relative movement of connected modules when commanded, thus refraining torque consumption when maintaining a robotic configuration due to gravity. This is beneficial for energy-saving reconfiguration in the modular robotic system. For modular robots without locking mechanism, the proposed methodology is still applicable, with modifications of energy consumption and subsequent derivation for heuristic planners.

With the experimental platform Mori, we present comparative simulation results using the proposed *automatic modeling algorithm* and *heuristic algorithm* respectively. Energy-optimal pre-folding patterns of reconfiguration are generated by enumeration with the *automatic modeling algorithm*. This process, however, is computationally expensive and may exceed the computing capabilities of on-board processors in robotic systems. To ease this problem, the reconfiguration planning can be implemented off-line before the transformation of modular robots. The *heuristic algorithm*, by contrast, yields optimal or near-optimal reconfiguration schemes with high quality, greatly reducing computation and time effort. Therefore, the *heuristic algorithm* is superior for real-time reconfiguration planning in autonomous environments, while the *automatic modeling algorithm* is suitable for modular systems with relatively few units in situations with strict limitation on energy expense.

We note that an important extension building upon our methods is reconfiguration planning with minimized peak torque in modular robots. Since energy consumption is closely related to the required joint torque, the energy-optimal planner can yield reconfiguration schemes with low peak torque. This indicates that either weaker actuators with lower torque limits are sufficient for a target shape with given modules, or larger modular architecture can be formed with given number of normal actuators. This is an important consideration with benefit for physical implementation in modular robots utilizing our methods. To further minimize the peak torque during the reconfiguration, more efforts are required to make corresponding modification of the model.

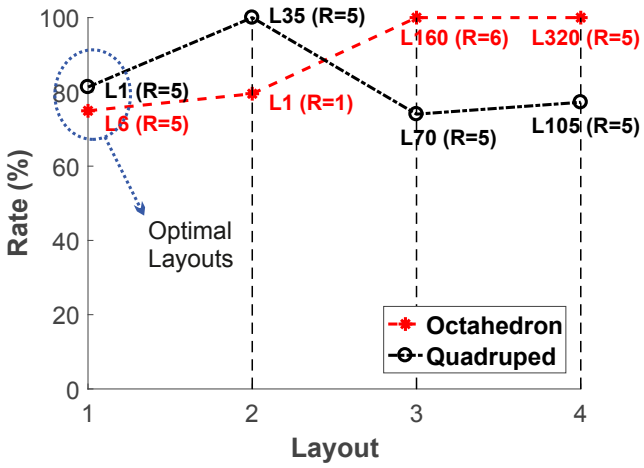


Figure 6. Monte-Carlo verification of the optimal folding sequence planner (Sec. 4.2.3). The first layouts are optimal layouts of Octahedron and Quadruped in Fig. 5, and the others are in Tab. 3 correspondingly. The rate evaluating performance of the folding sequence planner is computed as $\frac{K_1}{K} \times 100\%$, where K_1 is the number of randomly generated actuation orders, with larger torque consumption than that of the optimal folding sequence, and $K = 1000$ is the number of simulations with Monte-Carlo method.

The presented approach to reconfiguration using origami principles calls for more work on planning and control problems for modular robotic systems. This includes a promising extension that results in substantial cost and energy reductions with the use of passive modules, i.e., simple connection units without electronic components among the active ones. Developing an optimal distribution algorithm for active modules is essential in this problem for further energy saving purposes. Potential improvements also include theoretical investigation of computational complexity of initial patterns and folding sequence for shaping a desired 3D configuration. Further investigation can be conducted on collision avoidance to achieve energy-efficient folding free of collision combining our work. Another challenge that remains to be addressed is planning and control of the after transformation 3D configurations in different scenarios, such as gait generation algorithms for a four-legged locomotion robot, and distributed coordination control for multi-robot systems.

7 Conclusion

This paper presents a new approach to reconfiguration of modular robotic systems using origami principles that transform planar patterns into 3D shapes. The reconfiguration at this level significantly mitigates the misalignment problem by actuating modular units in controlled sequence rather than changing connectivity between modules. For the NP-complete problem of energy-optimal reconfiguration planning in modular robots, we established an algorithmic framework, including an automatic modeling algorithm and a heuristic algorithm. Both can generate energy-optimal reconfiguration schemes, for initial assembly and folding sequence of modules of a pre-folding pattern. The validity of the proposed heuristic algorithm is proved. The simulation results demonstrated on the Mori platform verified that the heuristic algorithm generates optimal or near-optimal schemes with computation efficiency, appropriate for real-time planning of energy-optimal reconfiguration for modular robots. Notably, the algorithmic framework developed in this paper can be applied to general modular robotic systems, for reconfiguration as origami folding is widely feasible with mechanical design. The work presented here should inspire algorithmic design for a wide range of problems in modular robotic systems in view of extensive and hybrid research overture.

Appendix. Proof of Theorem 1

1. Derivation of *leaf* module
   If module i is *leaf*, then $|C_i| = 0$, from Eq. 3 the dynamic terms f_i and n_i are written as

$$\begin{aligned} \boldsymbol{f}_i &= \boldsymbol{F}_i = m_i \ddot{\boldsymbol{r}}_i \\ \boldsymbol{n}_i &= \boldsymbol{N}_i + (\boldsymbol{p}_i + \boldsymbol{r}_i^*) \times \boldsymbol{F}_i \\ &= \boldsymbol{I}_i \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times (\boldsymbol{I}_i \boldsymbol{\omega}_i) + (\boldsymbol{p}_i + \boldsymbol{r}_i^*) \times \boldsymbol{F}_i \end{aligned} \tag{18}$$

Integrate $|\boldsymbol{f}_i|$ twice with time, Γ_i is written as

$$\begin{aligned} \Gamma_i &= \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |m_i \ddot{\boldsymbol{r}}_i| \, dtdt \\ &= m_i \left|\bar{\ddot{\boldsymbol{r}}}_i\right| (t_k - t_{k-1})^2 \\ &= m_i \kappa_1 \left| |\boldsymbol{r}_i(t_k)| - |\boldsymbol{r}_i(t_{k-1})| \right| \\ &= m_i \kappa_1 \left| \hat{r}_i(t_k) - \hat{r}_i(t_{k-1}) \right| \end{aligned} \tag{19}$$

where $\left|\bar{\ddot{\boldsymbol{r}}}_i\right|$ is average value of $|\ddot{\boldsymbol{r}}_i(t)|$ in time $t \in [t_{k-1}, t_k]$, and $\kappa_1 > 0$ is a constant.
Integrate $\boldsymbol{n}_i$ twice with time, Λ_i is written as

$$\begin{aligned} &\Lambda_i = \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{n}_i| \, dtdt \\ &\leq \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{I}_i \dot{\boldsymbol{\omega}}_i| \, dtdt + \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{\omega}_i \times (\boldsymbol{I}_i \boldsymbol{\omega}_i)| \, dtdt \\ &+ \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |(\boldsymbol{p}_i + \boldsymbol{r}_i^*) \times m_i \ddot{\boldsymbol{r}}_i| \, dtdt \end{aligned} \tag{20}$$

For each term in Eq. 20,

$$\begin{aligned} &\int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{I}_i \dot{\boldsymbol{\omega}}_i| \, dtdt \\ &\leq \|\boldsymbol{I}_i\| \cdot \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\dot{\boldsymbol{\omega}}_i| \, dtdt \\ &= \|\boldsymbol{I}_i\| \cdot \left|\bar{\dot{\boldsymbol{\omega}}}_i\right| (t_k - t_{k-1})^2 \\ &= \|\boldsymbol{I}_i\| \cdot \kappa_2 \left| |\boldsymbol{\theta}_i(t_k)| - |\boldsymbol{\theta}_i(t_{k-1})| \right| \end{aligned} \tag{21}$$

where $\left|\bar{\dot{\boldsymbol{\omega}}}_i\right|$ is average value of $|\dot{\boldsymbol{\omega}}_i(t)|$ in time $t \in [t_{k-1}, t_k]$ and $\kappa_2 > 0$ is a constant; and $\boldsymbol{\theta}_i(t)$ is the vector of generalized joint coordinates. The term

$$\begin{aligned} &\int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{\omega}_i \times (\boldsymbol{I}_i \boldsymbol{\omega}_i)| \, dtdt \\ &= \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} ||\boldsymbol{\omega}_i| \cdot |\boldsymbol{I}_i \boldsymbol{\omega}_i| \cdot \sin(\varphi_{i,1}) \cdot \hat{\boldsymbol{n}}_{i,1}| \, dtdt \\ &\leq \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{\omega}_i| \cdot |\boldsymbol{I}_i \boldsymbol{\omega}_i| \, dtdt \\ &\leq \|\boldsymbol{I}_i\| \cdot \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{\omega}_i|^2 \, dtdt \end{aligned} \tag{22}$$

where $\varphi_{i,1}$ and $\hat{\boldsymbol{n}}_{i,1}$ are vector angle and normal vector of $\boldsymbol{\omega}_i$ and $\boldsymbol{I}_i \boldsymbol{\omega}_i$.
To derive the term $\int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{\omega}_i|^2 \, dtdt$, reminding $\dot{q}_i$ as angular velocity of module i rotating about $\boldsymbol{z_i}$, and $|\dot{q}_i| = |\boldsymbol{\omega}_i|$, with unified joint motion planning (Sec. 3), it is deduced that

$$\dot{q}_i(\xi_1) \cdot \dot{q}_i(\xi_2) \geq 0, \ \xi_1, \ \xi_2 \in [t_{k-1}, t_k] \tag{23}$$

then it can be derived as

$$\begin{aligned} &\int_{t_{k-1}}^{t_k} |\boldsymbol{\omega}_i|^2 \, dt = \int_{t_{k-1}}^{t_k} |\dot{q}_i(t)|^2 \, dt \\ &= |\dot{q}_i(\xi)| \int_{t_{k-1}}^{t_k} |\dot{q}_i(t)| \, dt \\ &= |\dot{q}_i(\xi)| \cdot \left| |q_i(t_k)| - |q_i(t_{k-1})| \right|, \ \xi \in [t_{k-1}, t_k] \end{aligned} \tag{24}$$

and

$$\begin{aligned} &\int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{\omega}_i|^2 \, dtdt = \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\dot{q}_i(t)|^2 \, dtdt \\ &= |\dot{q}_i(\xi)| \cdot (t_k - t_{k-1}) \cdot \left| |q_i(t_k)| - |q_i(t_{k-1})| \right| \end{aligned} \tag{25}$$

Since $\dot{q}_i(\xi)$ is a constant with predefined joint motion, define $\bar{\omega}_i$ as average angular velocity of $\dot{q}_i$ in time $t \in [t_{k-1}, t_k]$, then

$$\begin{aligned} |\dot{q}_i(\xi)| \cdot (t_k - t_{k-1}) &= \kappa_3 \, |\bar{\omega}_i| \cdot (t_k - t_{k-1}) \\ &= \kappa_3 \, ||q_i(t_k)| - |q_i(t_{k-1})|| \end{aligned} \tag{26}$$

where κ_3 is a constant and $\kappa_3 > 0$. Therefore it is derived that

$$\int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{\omega}_i|^2 \, dtdt = \kappa_3 \, ||q_i(t_k)| - |q_i(t_{k-1})||^2 \tag{27}$$

then Eq. 22 can be expressed as

$$\begin{aligned} &\int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{\omega}_i \times (\boldsymbol{I}_i \boldsymbol{\omega}_i)| \, dtdt \\ &\leq \|\boldsymbol{I}_i\| \cdot \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{\omega}_i|^2 \, dtdt \\ &= \|\boldsymbol{I}_i\| \cdot \kappa_3 \, ||q_i(t_k)| - |q_i(t_{k-1})||^2 \end{aligned} \tag{28}$$

Since for each module, $|(\boldsymbol{p}_i + \boldsymbol{r}_i^*)|$ is constant as $|(\boldsymbol{p}_i + \boldsymbol{r}_i^*)| = \frac{l/2}{\sin(\pi/3)}$, and another term $\int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |(\boldsymbol{p}_i + \boldsymbol{r}_i^*) \times m_i \ddot{\boldsymbol{r}}_i| \, dtdt$ in Eq. 20 is derived as following, according to Eq. 19

$$\begin{aligned} &\int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |(\boldsymbol{p}_i + \boldsymbol{r}_i^*) \times m_i \ddot{\boldsymbol{r}}_i| \, dtdt \\ &= \left| |(\boldsymbol{p}_i + \boldsymbol{r}_i^*)| \cdot \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |m_i \ddot{\boldsymbol{r}}_i| \, dtdt \cdot \sin(\varphi_{i,2}) \cdot \hat{\boldsymbol{n}}_{i,2} \right| \\ &\leq \frac{l/2}{\sin(\pi/3)} m_i \kappa_1 \, |\hat{r}_i(t_k) - \hat{r}_i(t_{k-1})| \end{aligned} \tag{29}$$

where $\varphi_{i,2}$ and $\hat{\boldsymbol{n}}_{i,2}$ are vector angle and normal vector of $\boldsymbol{p}_i + \boldsymbol{r}_i^*$ and $\ddot{\boldsymbol{r}}_i$.
With $|\boldsymbol{\theta}_i(t)| = |q_i(t)|$ and the boundary conditions of $q_i(t_{k-1}) = 0$ and $q_i(t_k) = \hat{\theta}_i$, Λ_i^* defined as the maximum of Λ_i can be expressed as

$$\begin{aligned} \Lambda_i^* &= \left| \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} \boldsymbol{I}_i \dot{\boldsymbol{\omega}}_i \, dtdt \right| + \left| \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} \boldsymbol{\omega}_i \times (\boldsymbol{I}_i \boldsymbol{\omega}_i) \, dtdt \right| \\ &\quad + \left| \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} (\boldsymbol{p}_i + \boldsymbol{r}_i^*) \times m_i \ddot{\boldsymbol{r}}_i \, dtdt \right| \\ &= \|\boldsymbol{I}_i\| \cdot \kappa_2 \, |\boldsymbol{\theta}_i(t_k) - \boldsymbol{\theta}_i(t_{k-1})| + \|\boldsymbol{I}_i\| \cdot \kappa_3 \, ||q_i(t_k)| - |q_i(t_{k-1})||^2 \\ &\quad + \frac{l/2}{\sin(\pi/3)} m_i \kappa_1 \, |\hat{r}_i(t_k) - \hat{r}_i(t_{k-1})| \\ &= \|\boldsymbol{I}_i\| \cdot (\kappa_2 \left|\hat{\theta}_i\right| + \kappa_3 \left|\hat{\theta}_i\right|^2) + \frac{l/2}{\sin(\pi/3)} m_i \kappa_1 \, |\hat{r}_i(t_k) - \hat{r}_i(t_{k-1})| \end{aligned} \tag{30}$$

2. Derivation of non-*leaf* module
For modules that are not *leaves*, Γ_i and Λ_i are derived as

$$\begin{aligned} \Gamma_i &= \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} \left| \sum_{j=1}^{|\boldsymbol{C}_i|} \boldsymbol{f}_{\boldsymbol{C}_{i,j}} + \boldsymbol{F}_i \right| dtdt \\ &\leq \sum_{j=1}^{|\boldsymbol{C}_i|} \Gamma_{\boldsymbol{C}_{i,j}} + \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |m_i \ddot{\boldsymbol{r}}_i| \, dtdt \\ &\leq \sum_{j=1}^{|\boldsymbol{C}_i|} \Gamma_{\boldsymbol{C}_{i,j}}^* + m_i \kappa_1 \, |\hat{r}_i(t_k) - \hat{r}_i(t_{k-1})| \end{aligned} \tag{31}$$

thus Γ_i^* is written as

$$\Gamma_i^* = \sum_{j=1}^{|\boldsymbol{C}_i|} \Gamma_{\boldsymbol{C}_{i,j}}^* + m_i \kappa_1 \, |\hat{r}_i(t_k) - \hat{r}_i(t_{k-1})| \tag{32}$$

Since

$$\begin{aligned} &\int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} \left| \sum_{j=1}^{|\boldsymbol{C}_i|} \boldsymbol{p}_i \times \boldsymbol{f}_{\boldsymbol{C}_{i,j}} \right| dtdt \\ &\leq \sum_{j=1}^{|\boldsymbol{C}_i|} \left| |\boldsymbol{p}_i| \cdot \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} \left| \boldsymbol{f}_{\boldsymbol{C}_{i,j}} \right| dtdt \cdot \sin(\varphi_{i,j}) \cdot \hat{\boldsymbol{n}}_{i,j} \right| \\ &\leq \sum_{j=1}^{|\boldsymbol{C}_i|} |\boldsymbol{p}_i| \cdot \Gamma_{\boldsymbol{C}_{i,j}} \end{aligned} \tag{33}$$

where $\varphi_{i,j}$ and $\hat{\boldsymbol{n}}_{i,j}$ are vector angle and normal vector of $\boldsymbol{p}_i$ and $\Gamma_{\boldsymbol{C}_{i,j}}$.
Λ_i can be written as

$$\begin{aligned} \Lambda_i &\leq \sum_{j=1}^{|\boldsymbol{C}_i|} \Lambda_{\boldsymbol{C}_{i,j}} + \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} \left| \sum_{j=1}^{|\boldsymbol{C}_i|} \boldsymbol{p}_i \times \boldsymbol{f}_{\boldsymbol{C}_{i,j}} \right| dtdt \\ &\quad + \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{N}_i + (\boldsymbol{p}_i + \boldsymbol{r}_i^*) \times \boldsymbol{F}_i| \, dtdt \\ &\leq \sum_{j=1}^{|\boldsymbol{C}_i|} \Lambda_{\boldsymbol{C}_{i,j}} + \sum_{j=1}^{|\boldsymbol{C}_i|} |\boldsymbol{p}_i| \cdot \Gamma_{\boldsymbol{C}_{i,j}} \\ &\quad + \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\boldsymbol{N}_i + (\boldsymbol{p}_i + \boldsymbol{r}_i^*) \times \boldsymbol{F}_i| \, dtdt \\ &\leq \sum_{j=1}^{|\boldsymbol{C}_i|} (\Lambda_{\boldsymbol{C}_{i,j}}^* + |\boldsymbol{p}_i| \cdot \Gamma_{\boldsymbol{C}_{i,j}}^*) \\ &\quad + \|\boldsymbol{I}_i\| \cdot (\kappa_2 \left|\hat{\theta}_i\right| + \kappa_3 \left|\hat{\theta}_i\right|^2) \\ &\quad + \frac{l/2}{\sin(\pi/3)} m_i \kappa_1 \, |\hat{r}_i(t_k) - \hat{r}_i(t_{k-1})| \end{aligned} \tag{34}$$

Define $\lambda_1 = \|\boldsymbol{I}_i\|$, $\lambda_2 = \frac{l/2}{\sin(\pi/3)} m_i$, $\lambda_3 = m_i$, and $\lambda_4 = |\boldsymbol{p}_i| = 0$ or l, and the maximum of Λ_i can be expressed as

$$\begin{aligned} \Lambda_i^* &= \sum_{j=1}^{|\boldsymbol{C}_i|} (\Lambda_{\boldsymbol{C}_{i,j}}^* + \lambda_4 \cdot \Gamma_{\boldsymbol{C}_{i,j}}^*) + \lambda_1 \cdot (\kappa_2 \left|\hat{\theta}_i\right| + \kappa_3 \left|\hat{\theta}_i\right|^2) \\ &\quad + \lambda_2 \kappa_1 \, |\hat{r}_i(t_k) - \hat{r}_i(t_{k-1})| \end{aligned} \tag{35}$$

with the maximum of Γ_i written as

$$\Gamma_i^* = \sum_{j=1}^{|\boldsymbol{C}_i|} \Gamma_{\boldsymbol{C}_{i,j}}^* + \lambda_3 \kappa_1 \, |\hat{r}_i(t_k) - \hat{r}_i(t_{k-1})| \tag{36}$$

Therefore, the estimation of torque consumption $\varepsilon_i' = \int_{t_{k-1}}^{t_k} \int_{t_{k-1}}^{t_k} |\tau_i| \, dtdt$ can be represented by Λ_i^* as in Eq. 35.

Acknowledgements

This work is sponsored by the National Natural Science Foundation of China (NSFC) through grants No. 61503102 and No. 61673057, and partially by the Swiss National Science Foundation (SNSF) START project. We thank the above mentioned funds for their financial support.

References

An B, Miyashita S, Tolley MT, Aukes DM, Meeker L, Demaine ED, Demaine ML, Wood RJ and Rus D (2014) An end-to-end approach to making self-folded 3d surface shapes by uniform heating. In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, pp. 1466–1473.

An B and Rus D (2014) Designing and programming self-folding sheets. *Robotics and Autonomous Systems* 62(7): 976–1001.

Balkcom DJ and Mason MT (2008) Robotic origami folding. *The International Journal of Robotics Research* 27(5): 613–627.

Belke CH and Paik J (2017) Mori: A modular origami robot. *IEEE/ASME Transactions on Mechatronics* .

Bi Z and Zhang WJ (2001) Concurrent optimal design of modular robotic configuration. *Journal of Robotic systems* 18(2): 77–87.

Casal A and Yim MH (1999) Self-reconfiguration planning for a class of modular robots. In: *Photonics East'99*. International Society for Optics and Photonics, pp. 246–257.

Chen IM and Yang G (1998) Automatic model generation for modular reconfigurable robot dynamics. *Journal of dynamic systems, measurement, and control* 120(3): 346–352.

Chen IM, Yeo SH, Chen G and Yang G (1999) Kernel for modular robot applications: automatic modeling techniques. *The International Journal of Robotics Research* 18(2): 225–242.

Chen Y, Peng R and You Z (2015) Origami of thick panels. *Science* 349(6246): 396–400.

Fei Y, Zhao X and Xu W (1998) Kinematics and dynamics of reconfigurable modular robots. In: *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, volume 4. IEEE, pp. 3325–3334.

Filipov ET, Tachi T and Paulino GH (2015) Origami tubes assembled into stiff, yet reconfigurable structures and metamaterials. *Proceedings of the National Academy of Sciences* 112(40): 12321–12326.

Firouzeh A and Paik J (2015) Robogami: a fully integrated low-profile robotic origami. *Journal of Mechanisms and Robotics* 7(2): 021009.

Guan Y, Yokoi K, Stasse O and Kheddar A (2005) On robotic trajectory planning using polynomial interpolations. In: *Robotics and Biomimetics (ROBIO). 2005 IEEE International Conference on*. IEEE, pp. 111–116.

Guzman R, Navarro R, Ferre J and Moreno M (2016) Rescuer: Development of a modular chemical, biological, radiological, and nuclear robot for intervention, sampling, and situation awareness. *Journal of Field Robotics* 33(7): 931–945.

Ha S, Coros S, Alspach A, Kim J and Yamane K (2017) Joint optimization of robot design and motion parameters using the implicit function theorem. In: *Proceedings of Robotics: Science and Systems*. Cambridge, Massachusetts. DOI:10.15607/RSS.2017.XIII.003.

Haenselmann T and Effelsberg W (2012) Optimal strategies for creating paper models from 3d objects. *Multimedia systems* 18(6): 519–532.

Hollerbach JM (1980) A recursive lagrangian formulation of maniputator dynamics and a comparative study of dynamics formulation complexity. *IEEE Transactions on Systems, Man, and Cybernetics* 10(11): 730–736.

Hou F and Shen WM (2010) On the complexity of optimal reconfiguration planning for modular reconfigurable robots. In: *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, pp. 2791–2796.

Hou F and Shen WM (2014) Graph-based optimal reconfiguration planning for self-reconfigurable robots. *Robotics and Autonomous Systems* 62(7): 1047–1059.

Kamimura A, Kurokawa H, Yoshida E, Murata S, Tomita K and Kokaji S (2005) Automatic locomotion design and experiments for a modular robotic system. *IEEE/ASME Transactions on mechatronics* 10(3): 314–325.

Kurokawa H, Tomita K, Kamimura A, Kokaji S, Hasuo T and Murata S (2008) Distributed self-reconfiguration of m-tran iii modular robotic system. *The International Journal of Robotics Research* 27(3-4): 373–386.

Lang RJ (1996) A computational algorithm for origami design. In: *Proceedings of the twelfth annual symposium on Computational geometry*. ACM, pp. 98–105.

Liu J, Wang Y, Ma S and Li Y (2010) Enumeration of the non-isomorphic configurations for a reconfigurable modular robot with square-cubic-cell modules. *International Journal of Advanced Robotic Systems* 7(4): 31.

Meister E, Gutenkunst A and Levi P (2013) Dynamics and control of modular and self-reconfigurable robotic systems. *International Journal on Advances in Intelligent Systems Volume 6, Number 1 & 2, 2013* .

Paik JK, Byoungkwon A, Rus D and Wood RJ (2012) Robotic origamis: Self-morphing modular robot. In: *ICMC*, EPFL-CONF-206919.

Park FC and Bobrow JE (1994) A recursive algorithm for robot dynamics using lie groups. In: *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*. IEEE, pp. 1535–1540.

Peraza-Hernandez EA, Hartl DJ, Malak Jr RJ and Lagoudas DC (2014) Origami-inspired active structures: a synthesis and review. *Smart Materials and Structures* 23(9): 094001.

Romanishin JW, Gilpin K and Rus D (2013) M-blocks: Momentum-driven, magnetic modular robots. In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, pp. 4288–4295.

Straub R and Prautzsch H (2011) *Creating optimized cut-out sheets for paper models from meshes*. KIT, Fakultät für Informatik.

Tachi T (2009) Simulation of rigid origami. *Origami* 4: 175–187.

Tachi T (2010) Origamizing polyhedral surfaces. *IEEE transactions on visualization and computer graphics* 16(2): 298–311.

Wang L, Plecnik MM and Fearing RS (2016) Robotic folding of 2d and 3d structures from a ribbon. In: *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, pp. 3655–3660.

Wang W, Dong W, Su Y, Wu D and Du Z (2014) Development of search-and-rescue robots for underground coal mine applications. *Journal of Field Robotics* 31(3): 386–407.

Watanabe N and Kawaguchi Ki (2009) The method for judging rigid foldability. *Origami* 4: 165–174.

Wei H, Cai Y, Li H, Li D and Wang T (2010) Sambot: A self-assembly modular robot for swarm robot. In: *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, pp. 66–71.

Wu W, Guan Y, Yang Y and Dong B (2016) Multi-objective configuration optimization of assembly-level reconfigurable modular robots. In: *Information and Automation (ICIA), 2016*

IEEE International Conference on. IEEE, pp. 528–533.

Yang G and Chen IM (2000) Task-based optimization of modular robot configurations: minimized degree-of-freedom approach. *Mechanism and machine theory* 35(4): 517–540.

Yao M, Cui H, Xiao X, Belke CH and Paik J (2018a) Towards peak torque minimization for modular self-folding robots. In: *Intelligent Robots and Systems (IROS), 2018 IEEE/RSJ International Conference on*. IEEE.

Yao M, Xiao X, Belke CH, Cui H and Paik J (2018b) Optimal distribution of active modules in reconfiguration planning of modular robots. *Journal of Mechanisms and Robotics* .

Yao M, Xiao X, Tian Y and Cui H (2018c) A two-time scale control scheme for on-orbit manipulation of large flexible module. *Acta Astronautica* .

Zhakypov Z, Belke C and Paik J (2017) Tribot: A deployable, self-righting and multi-locomotive origami robot. In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*, EPFL-CONF-230280.

Zhang H, Wang W, Deng Z, Zong G and Zhang J (2006) A novel reconfigurable robot for urban search and rescue. *International Journal of Advanced Robotic Systems* 3(4): 48.

Zirbel SA, Lang RJ, Thomson MW, Sigel DA, Walkemeyer PE, Trease BP, Magleby SP and Howell LL (2013) Accommodating thickness in origami-based deployable arrays. *Journal of Mechanical Design* 135(11): 111005.

Zykov V, Mytilinaios E, Desnoyer M and Lipson H (2007) Evolved and designed self-reproducing modular robotics. *IEEE Transactions on robotics* 23(2): 308–319.