IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 12, 2018, accepted December 15, 2018, date of publication December 19, 2018,
date of current version January 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2888588

A Self-Reconfiguration Planning Strategy for Cellular Satellites

YIZHAI ZHANG, (Member, IEEE), WENHUI WANG, JINGHAN SUN, HAITAO CHANG, AND PANFENG HUANG, (Senior Member, IEEE)
School of Astronautics, Northwestern Polytechnical University, Xi'an 710072, China
National Key Laboratory of Aerospace Flight Dynamics, Northwestern Polytechnical University, Xi'an 710072, China

Corresponding author: Panfeng Huang (pfhuang@nwpu.edu.cn)

This work was supported in part by the National Science Fund for Distinguished Young Scholars of China under Grant 61725303, in part by the National Natural Science Foundation of China under Grant 61873204, and in part by the Chinese Fundamental Research Funds for the Central Universities under Award 3102017jg02005.

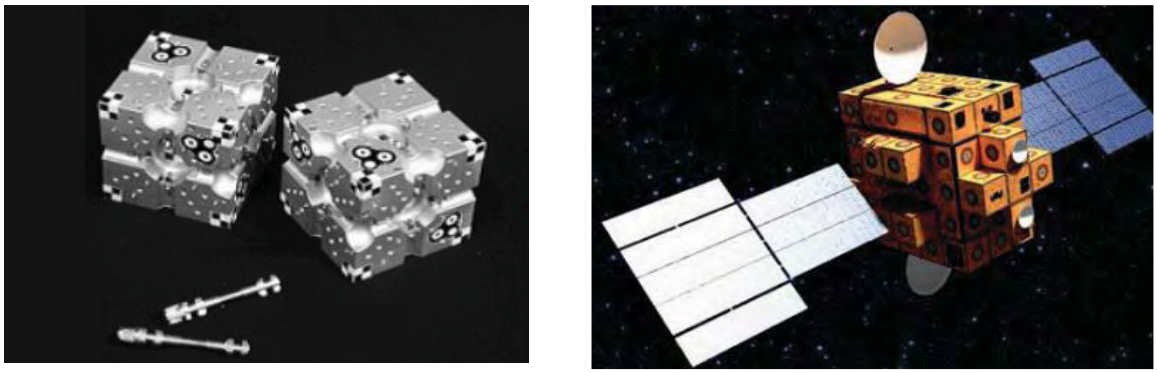
ABSTRACT Cellular satellites, which disaggregate space systems into many small and simple unit cells, represent a promising direction in space system construction. Cellular satellites can greatly change the design of space systems with attractive features, such as rapid response, task flexibility, and a high effectiveness-to-cost ratio. On-orbit self-reconfiguration is critical to enabling cellular satellites to fulfill their potential. In this paper, a self-reconfiguration planning strategy is proposed for cellular satellites. Using this strategy, the cellular satellite can autonomously evolve its morphology from an initial configuration to an expected configuration, mainly by the assistance of assembling cell. The assembling cell design and the detailed self-reconfiguration planning algorithms are presented in this paper. The extensive visual simulations are conducted to validate the effectiveness of the proposed strategy.

INDEX TERMS Cellular satellite, self-reconfiguration, planning, assembling cell, robotic arm.

I. INTRODUCTION

Along with large-scale exploitation of outer space, spacecraft and satellites are being launched increasingly often. However, traditional designs of space systems cannot satisfy future requirements [1]–[3]. For example, future space systems are expected to possess low cost, rapid response, and multiple purposes [4]. However, the current space system design generally requires subsystem-by-subsystem development and extensive iterations, which are slow, repetitive and expensive [5], [6]. Due to a long development period and launch window constraints, it is also difficult for current space systems to respond rapidly to emergency situations. Moreover, space systems are usually designed with specific missions and thus, cannot adapt to unplanned tasks.

To solve the limitations of current space systems, several new concepts have been proposed recently, such as Cubesats [7], [8], satellite formation [9], [10], separate modular spacecraft [11], [12], and cellular satellites [13], [14]. Among them, cellular satellites are promising and have attracted considerable attention. The cellular satellite is a new means of space system construction that is disaggregated into many small and simple unit cells. All unit cells are standardized, physically separated, functionally independent, and interconnected with each other through standard interfaces. These unit cells are launched into space in advance and are then assembled when a space mission is assigned. Once the mission changes, these cells can be re-configured on-orbit to another space system for a new mission. Cellular satellites have many attractive features, such as rapid response, task flexibility, and a high effectiveness-to-cost ratio, which alters space system design and operating modes. Hideyuki Tanaka first proposed the concept of cellular satellites in 2006 and designed several unit cell prototypes and developed a standard mechanical interface (Fig. 1(a)) [15]. Recently, DLR (Deutsches Zentrum fur Lüft-und Raumfahrt, GER) and DARPA (Defense Advanced Research Projects Agency, USA) carried out the iBOSS (Intelligent Building Blocks for On-Orbit-Satellite Servicing) plan [16], [17] and the Phoenix Program [18], [19], respectively. DLR proposed breaking down a conventional satellite bus into single standardized, intelligent building blocks for on-orbit-satellite servicing (Fig. 1(b)). The relevant subsystem components are accommodated inside each building block. The Phoenix Program released the ''satlet'' concept as a satellite architectural unit. Each satlet facilitates some portion of the overall functions that provide the space system all required



(a) (b)

FIGURE 1. Examples of cellular satellites. (a) CellSat. (b) IBOSS.

(a) (b)

FIGURE 2. (a) Cellular satellite developed by NWPU. (b) Prototypes of two unit cells.

capabilities when aggregated via hardware and software. NWPU (Northwestern Polytechnical University, China) put forth the concept of heterogeneous cellular satellite, developed cell prototypes and studied the problems of inertia parameters identification (Fig. 2) [20], [21].

The most attractive feature of cellular satellite is the on-orbit self-reconfiguration capacity. Since research on cellular satellites began to emerge, most work has focused on the prototype design of unit cells and few studies have investigated on-orbit self-reconfiguration strategies, algorithms and techniques. The self-reconfiguration concept in the cellular satellite is similar to that of ground modular robots [22], such as M-tran robot [23] and UBot [24]. However, the self-reconfiguration strategy of ground modular robotics cannot be applied directly to cellular satellite for several reasons. First, in ground modular robots, each module is usually assumed to have strong sensing and locomotion abilities [22]. For cellular satellites, however, most unit cells are simple with single functions and thus cannot move without external assistance. Second, ground modular robots are usually homogeneous in that all modules are the same or similar, whereas cellular satellites are completely heterogeneous [25]. Third, studies of ground modular robots have focused on planning coordinated motions of modules and implementing efficient self-adaptive transformation in complex and hostile environments [26]. Yet the cellular satellites emphasize how to construct a serviceable space system. Given the obvious design and operational differences between space systems and ground systems, key problems in self-reconfiguration for ground modular robots and cellular satellites are totally different. Extensive self-reconfiguration studies for cellular satellites remain lacking.

In this paper, we propose a self-reconfiguration planning strategy for cellular satellites, under which the cellular satellite can evolve its morphology gradually from an arbitrary initial configuration to an arbitrary expected configuration. The contributions of this work are twofold. First, we study a real and critical issue in the use of cellular satellites. To the

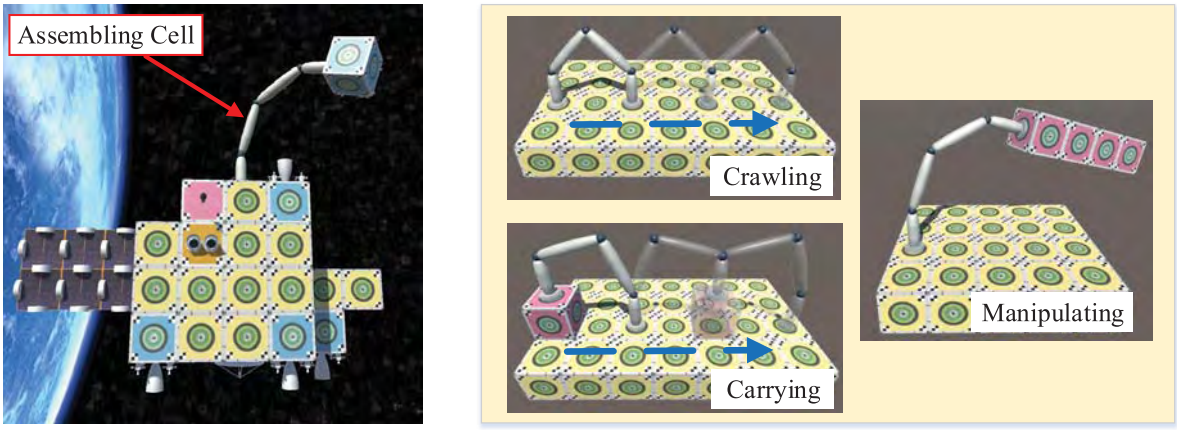


FIGURE 3. (a) Self-reconfiguration using assembling cell. (b) Typical motion modes of assembling cell (crawling, carrying a payload cell, and manipulating multiple cells).

best of the authors' knowledge, few studies have investigated the planning problem of on-orbit self-reconfiguration in cellular satellites. Second, this study can be extended to a class of modular robots whose modules are heterogeneous and where only partial modules have mobility.

The remainder of this paper is organized as follows. Section II presents the problem description of self-reconfiguration planning of cellular satellites. Section III introduces the assembling cell design. The details of self-reconfiguration algorithms are discussed in Section IV. Extensive simulations are used to validate the proposed strategy in Section V. Section VI offers concluding remarks.

II. PROBLEM DESCRIPTION

The immediate target of this study is a cellular satellite designed by NWPU. The cellular satellite consists of many types of unit cells (Fig. 2) that can be divided into two classes. One class comprises basic cells, which are strict cubes with six identical surfaces. Each surface has one standard interface in the center. Even though basic cells have the same appearance, they may have different intrinsic functions; this class includes controller cells, computation cells, and navigation cells, etc. The other class represents specialized cells, such as thruster cells, communication cells, and solar panel cells. At least one cell surface is unique in this class; for example, thruster cells have a nozzle module on one surface and a communication cell is mounted with an antenna. The cellular satellite is a completely heterogeneous system.

On-orbit self-reconfiguration indicates that the cellular satellite can gradually change its morphology step-by-step, without relying on astronauts or a large and expensive space robotic platform. Although on-orbit self-reconfiguration is critical for cellular satellites, its planning problem is complicated due to the following reasons and constraints:

- Most unit cells do not have movement ability, because an essential feature of the cellular satellite is that unit cells are singly functional;
- Any unit cell is not allowed to be separated from the main satellite body. In other words, all unit cells must be interconnected at all times to prohibit free floating in a zero-gravity environment;
- The move sequences and placement locations of unit cells are important. For example, outer unit cells must be disassembled first before moving inner unit cells. The locations of current unit cells cannot obstruct those planned to move in the next step;
- A minimum number of cell moves is always preferred for on-orbit reconfiguration of cellular satellites.

In the cellular satellite developed by NWPU, a special cell, called the assembling cell, is responsible for moving or manipulating unit cells Fig. 3(a). The assembling cell is a specially-made symmetric robotic arm, and each end has one standard interface. Thus, the assembling cell can flexibly move on free interfaces by alternating ''head'' and ''foot'', similar to crawling robots. In addition, the assembling cell is capable of carrying one payload cell to a new place and can stand still to manipulate a combination of multiple cells. The three typical motion modes of the assembling cell are shown in Fig. 3(b). With the help of the assembling cell, the cellular satellite can obtain on-orbit self-reconfiguration capability.

In this paper, we present a self-reconfiguration planning strategy for cellular satellites using the assembling cell. Several assumptions are adopted in this study to specify the problem. First, only one assembling cell is used in the self-reconfiguration process. Second, the number of unit cells in the initial configuration is the same as that in the expected configuration. Third, except solar panel cells, all other cells have at most one unique side with functional components (Fig. 2(a)). Solar panel cells have four sides with standard interfaces (Fig. 2(a)). Fourth, we do not consider the orbital dynamics effect and other influences of space environment during self-reconfiguration (e.g., orbit, zero gravity) and focus only on top-level planning problem.

Remark 1: In this study, we do not consider the influences of space environment for the following reasons. First, the study focuses on the top-level planning problem during self-reconfiguration rather than the dynamics and control side. Considering page limits, we do not report every aspects in one paper. Second, the robotic manipulations in space have been extensively studied in literature and we do not repeat other work. Third, the movement of each unit cell on cellular satellites is just relative motion and compared to the entire cellular satellites, the mass of each unit cell is also limited. Therefore, we can simplify the planning problem by neglecting external influences (e.g., orbit, zero gravity).

III. DESIGN OF ASSEMBLING CELL

The assembling cell plays an important role in self-reconfiguration. In this work, we propose a four-link symmetrical robotic arm as the assembling cell (Fig. 4). The assembling cell has five rotational joints: three are at the intersections between two adjacent links, and two exist inside the two ends. The assembling cell can move freely on the surface of the cellular satellite by utilizing free interfaces as footholds. The size of the assembling cell is proportional to the side of the unit cell. Provided that the side length of a basic cell is 1 unit, the lengths of the four links of the assembling cell are 1 unit, 1.5 units, 1.5 units, and 1 unit, respectively. To facilitate presentation in the rest of this paper, we define six coordinate frames $\mathcal{F}_0(x_0, y_0, z_0) \cdots \mathcal{F}_5(x_5, y_5, z_5)$ with the origins at two ends and four links, respectively; see Fig. 4. By default, we assume $\mathcal{F}_0$ presents the base coordinate frame fixed with the main body cellular satellite and $\mathcal{F}_5$ at the moving end. Once "head" and "foot" are alternated during crawling or carrying motions, the coordinate system is mirrored. Given the coordinate frames, we denote the corresponding joint rotational angles as $\theta_1 \cdots \theta_5$. In this case, θ_1 and θ_5 are rotated with z_0 and z_4, respectively, and θ_2, θ_3 and θ_4 are rotated with x_1, x_2 and x_3, respectively. Note that x_1, x_2 and x_3 are always parallel with each other.

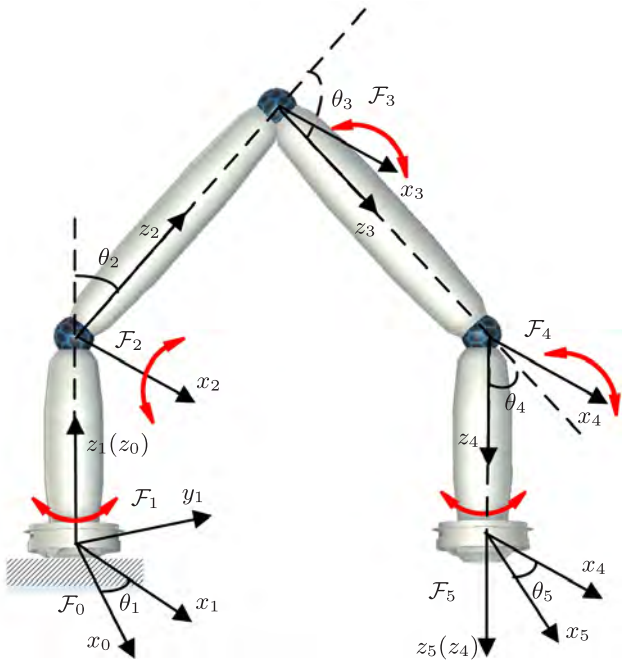


FIGURE 4. Assembling cell and its coordinate system.

Fig. 5 shows sample operating conditions of the assembling cell during crawling and carrying motions. Green blocks represent the "foot" of the assembling cell and red blocks represent where another end can reach (for carrying motion, the end is extended to the side of the payload cell). We do not show all operating conditions of the assembling cell due to the large total number (seventeen for crawling motion and nine for carrying motion). But all operating conditions constitute two workplace sets of the assembling cell, denoted as $\mathcal{W}_w$ for crawling motion and $\mathcal{W}_c$ for carrying motion, respectively.

IV. DESIGN OF SELF-RECONFIGURATION PLANNING ALGORITHM

A. OVERALL ALGORITHM DESCRIPTION

In this study, the self-reconfiguration planning algorithm is divided into three core sub-algorithms: task planning, path planning and joint planning; see Fig. 6.

1) TASK PLANNING

Task planning decomposes the morphology transformation of the cellular satellite into continuous movements of individual unit cells; that is, it calculates the move sequences and placement positions of unit cells. After task planning, the starting position and terminal position for each unit cell movement are determined.

2) PATH PLANNING

This algorithm finds feasible intermediate footholds on the surface for crawling or carrying motions of the assembling cell. It is usually impossible for the assembling cell to move or carry a payload cell between the starting position and terminal position within one step due to its limited length.

3) JOINT PLANNING

Joint planning calculates the joint rotational angles (i.e., $\theta_1 \cdots \theta_5$) to realize a specific motion of the assembling cell, such as a one-step crawling motion or one-step carrying motion.

By combining these sub-algorithms, we obtain an integrated algorithm for the cellular satellite to achieve self-reconfiguration capacity.

B. TASK PLANNING

Inspired by the work in [27], we propose a four-step task planning algorithm consisting of classifying, melting, sorting and growing; see Fig. 7. The basic idea is to bridge the initial and expected configurations via an intermediate configuration. In the classifying step, we compute and design

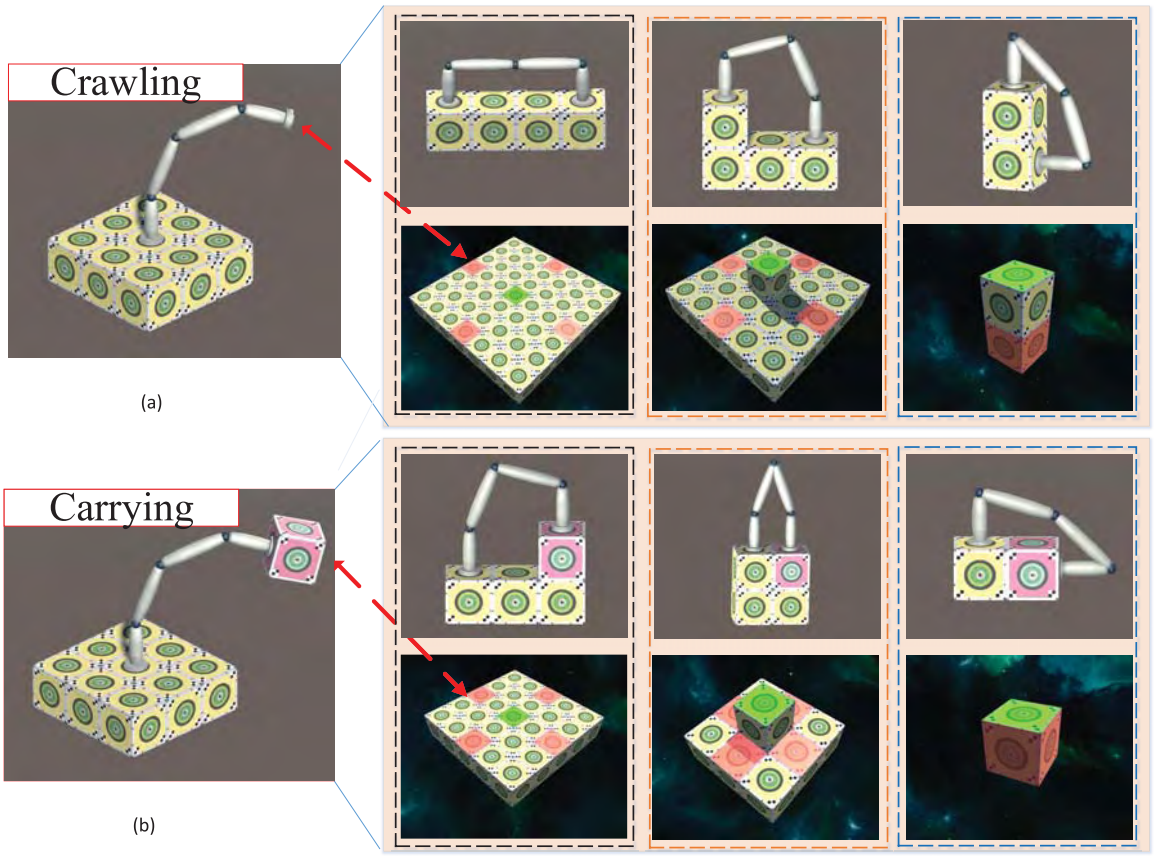


FIGURE 5. Examples of operating conditions of assembling cell. (a) Crawling motion. (b) Carrying motion.

FIGURE 6. Self-reconfiguration algorithm.

an appropriate intermediate configuration. In the melting and sorting steps, the cellular satellite transforms its morphology from an initial configuration to an intermediate configuration cell-by-cell. During the growing step, the expected configuration is then constructed from the intermediate configuration cell-by-cell. We denote the initial configuration, intermediate configuration, and expected configuration as $\mathcal{O}$, $\mathcal{I}$, and $\mathcal{E}$, respectively. We number all cells with C_k, where $k = 1 \cdots N$ and N represents the total number of unit cells (Fig. 7).

In this work, although the initial configuration and expected configuration are arbitrary three-dimensional (3D) structures, the intermediate configuration is always designed as a 2D structure for simplicity. For clarity of presentation, we use a 2D example for the initial configuration and expected configuration in the schematics in this section; see Fig. 7.

Remark 2: In the proposed algorithm, task planning focuses only on placement of unit cells. The orientation adjustment for each unit cell between the initial and expected

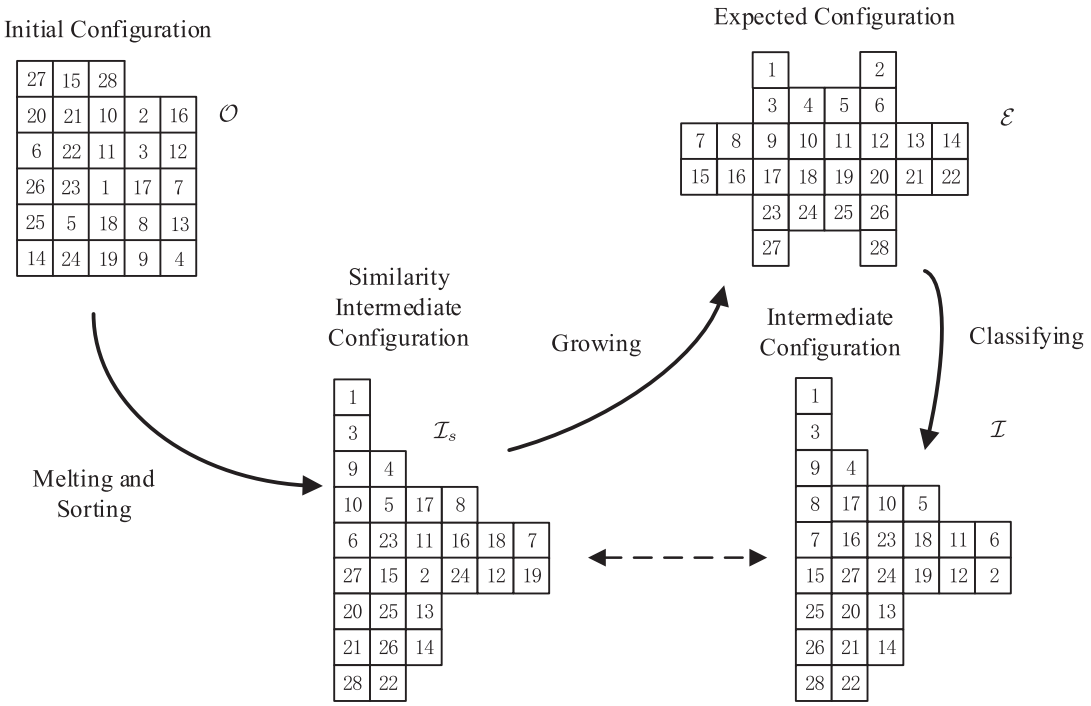


FIGURE 7. Diagram of proposed four-step task planning algorithm (initial configuration and expected configuration are 3D structures).

configurations will be considered and explained in path planning and joint planning.

Remark 3: Note that in Fig. 7, we actually use similarity intermediate configuration as the bridge between initial and expected configurations rather than intermediate configuration. We will explain this in melting and sorting steps.

1) CLASSIFYING

The purpose of the classifying step is to design an appropriate intermediate configuration $\mathcal{I}$ by analyzing expected configuration $\mathcal{E}$. All cells in $\mathcal{I}$ are temporarily placed with a ready order, and we can easily construct $\mathcal{E}$ cell-by-cell without adjusting any cell's position. Algorithm 1 illustrates the classifying process. To achieve this effect, we first analyze the connection topology of expected configuration $\mathcal{E}$ and attribute all cells with different assembling priorities. We denote the combination of cells with the same priority number i as $\boldsymbol{l}_i$. Cells with a smaller assembling priority number must be settled in $\mathcal{E}$ before those with lager assembling priority number. For instance, the cell in $\boldsymbol{l}_2$ should be moved and settled before the cell in $\boldsymbol{l}_4$. Clearly, $\boldsymbol{l}_0$ represents the first settled cell. Apart from $\boldsymbol{l}_0$, other $\boldsymbol{l}_i$ may have multiple cells. The move sequences of cells with the same assembling priority number are interchangeable. Therefore, we can obtain a convenient design of $\mathcal{I}$ by simply stacking $\boldsymbol{l}_i$ in order. The problem then concerns how to generate $\boldsymbol{l}_i$. In this study, we use the breadth-first search method to analyze expected configuration $\mathcal{E}$ and to introduce a tree form (denoted as $\boldsymbol{T}_e$); see Fig. 8. Therefore, $\boldsymbol{l}_0$ is the root node of $\boldsymbol{T}_e$ and $\boldsymbol{l}_i$ corresponds to all child nodes in the i-th layer of $\boldsymbol{T}_e$. Clearly, the choice of the root node cell is not unique and $\boldsymbol{T}_e$ is directly related to $\boldsymbol{l}_0$. In Algorithm 1, we traverse all the cells on the surface of $\mathcal{E}$ (the cell set containing all surface cells is denoted as $\mathcal{S}_e$) and choose one that generates the fewest layers.

Algorithm 1: Classifying Step

```
Input: Expected Configuration E ;
for C_k ∈ S_e do
    Generate tree form from C_k via breadth-first search method ;
end
T_e ← Tree with fewest layers ;
l_i is the combination of all child nodes on i-th layer of T_e ;
Generate I by stacking l_i in order;
Output: Intermediate Configuration I ;
```

2) MELTING AND SORTING

The aim of the melting and sorting steps is to disassemble initial configuration $\mathcal{O}$ and construct intermediate configuration $\mathcal{I}$; see Algorithm 2. Because no unit cell is allowed to completely separated from the main body of the cellular satellite, $\mathcal{I}$ is constructed from one free interface of one surface unit cell of $\mathcal{O}$. We denote the surface unit cell as C_s. In this study, C_s can be chosen arbitrarily; we choose one that can help to reduce collision risk in the self-reconfiguration process, such as cells on the convex surface. C_s is the junction

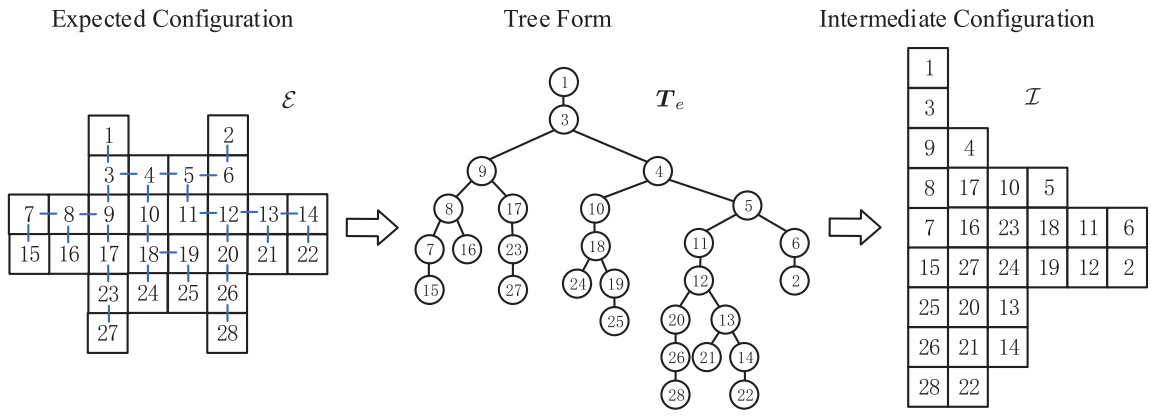


FIGURE 8. Diagram of classifying algorithm.

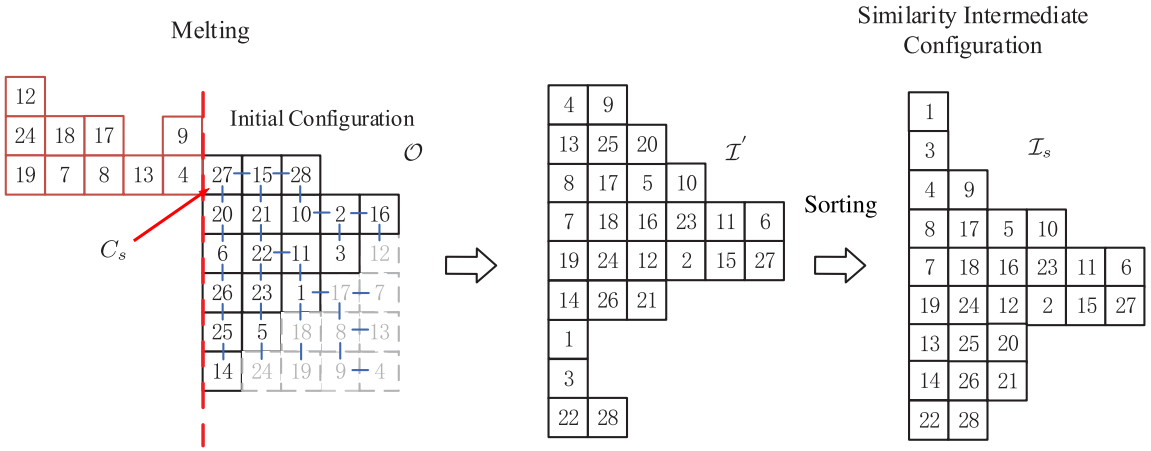


FIGURE 9. Diagram of melting and sorting algorithms.

Algorithm 2: Melting and Sorting Steps

```
Input: Initial Configuration O ;
Choose a junction node C_s on surface of O ;
Generate a tree form originating from C_s;
for k = 1 ··· N do
    Move the farthest child node of the tree ;
    if C_i1 ∈ l_i then
        Consider as basis cell and align with other basis
        cells to form a chain
    else
        Aligned with other cells in l_i to from a chain
    end
end
Obtain I' (a combination of l_i);
Sorting I' and obtain I_s ;
Output: Similarity Intermediate Configuration I_s;
```

between $\mathcal{O}$ and $\mathcal{I}$; see Fig. 9. Similar to the classifying step, to disassemble $\mathcal{O}$ more efficiently, we search a tree form in $\mathcal{O}$ originating from C_s and disassemble the tree gradually from the farthest child nodes. In addition, all cells in $\mathcal{O}$ are assigned by their assembling priorities (i.e., $\boldsymbol{l}_i$) determined in the classifying step. During the melting process, we adopt a first-come-first-settle rule and define the first arriving cell in each $\boldsymbol{l}_i$ as the basis cell, denoted by C_{i1}. All basis cells for different priorities form a chain structure one-by-one originating from the free interface of C_s. Cells within the same $\boldsymbol{l}_i$ are aligned one-by-one originating from their basis cell to form a chain structure, which is somehow perpendicular to the chain with basis cells. We denote the 2D configuration after the melting step as $\mathcal{I}'$ (Fig. 9).

Clearly, $\mathcal{I}'$ is also a combination of $\boldsymbol{l}_i$. Comparing $\mathcal{I}'_i$ with $\mathcal{I}$, we can observe two differences. First, the orders of $\boldsymbol{l}_i$ are different. Second, within each $\boldsymbol{l}_i$, the sequences of unit cells are also different. Therefore, we next sort the order of $\boldsymbol{l}_i$ in $\mathcal{I}'$ and obtain a new configuration form $\mathcal{I}_s$. As all unit cells in each $\boldsymbol{l}_i$ are connected to their basis cell, we only need to sort the order of basis cells by using multiple cell manipulation of the assembling cell. This phase can be conducted by many sorting methods, such as the traditional exchange sort method and bubble sort method. For the second difference, as mentioned previously, the move sequences of

cells within the same $\boldsymbol{l}_i$ are interchangeable and therefore, the difference in $\boldsymbol{l}_i$ has no influence on the construction of expected configuration $\mathcal{E}$. We consider configuration $\mathcal{I}_s$ to be a similarity configuration to $\mathcal{I}$ (Fig. 9). Accordingly, we use $\mathcal{I}_s$ instead of $\mathcal{I}$ in the final growing step.

Remark 4: During the sorting step, one practical problem involves how to break, insert and sort the intermediate configuration by using only one assembling cell. In this study, because the assembling cell can manipulate multiple cells one time (e.g., all cells in one $\boldsymbol{l}_i$), we use a similar treatment for the Tower of Hanoi problem. Details of the sorting step when using one assembling cell are not discussed because they are trivial, tedious, and not the main contribution of this work. An example of this process is presented in the simulation section. In addition, this difficulty can be easily addressed by using multiple assembling cells.

3) GROWING

After the sorting step, all cells in $\mathcal{I}_s$ are placed in the appropriate order and therefore, $\mathcal{E}$ can be constructed cell-by-cell readily and conveniently; see Fig. 10. $\boldsymbol{l}_0$ is the first settled cell.

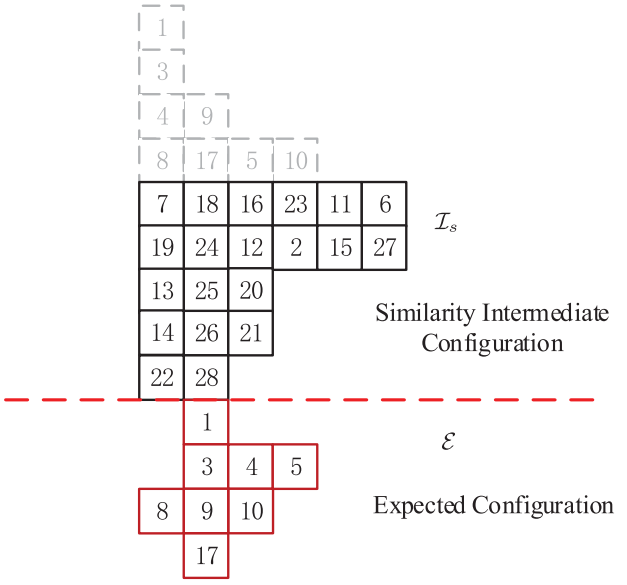


FIGURE 10. Diagram of growing algorithm.

C. PATH PLANNING

Due to the length constraints of the assembling cell, it is usually impossible for assembling cell to complete a crawling operation or carrying operation within one step. We denote the first interface and last interface as P_s and P_t, respectively. The idea of path planning is to identify the shortest and successive footholds on the surface of the cellular satellite from P_s to P_t by using $\mathcal{W}_w$ or $\mathcal{W}_c$ as workspace sets. We denote these successive footholds as path $\boldsymbol{\tau}$. The details of the path planning algorithm are presented in Algorithm 3.

Algorithm 3: Path Planning

```
Input: First interface P_s and last interface P_t ;
Initialize a queue Q and add P_s into Q;
while P_t ∉ Q do
    Current interface P_c ← Get the next element in Q ;
    Search reachable interfaces from P_c by W_w and W_c ;
    Set the backpointers from reachable interfaces to P_c;
    Add reachable interfaces into Q;
end
Generate a path τ from P_t to P_s through back
backpointers ;
Output: Path τ;
```

Inspired by the Dijkstra algorithm [28], we use a searching-based method to find $\boldsymbol{\tau}$ step-by-step. We first define a queue $\boldsymbol{Q}$ to include interfaces that have been searched already. Once P_t is reached, backpointers are used to trace $\boldsymbol{\tau}$; see Fig. 11.

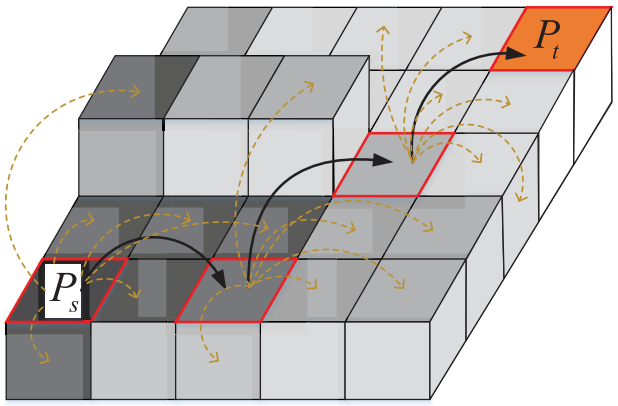


FIGURE 11. Diagram of path planning.

For the crawling operation, P_s and P_t are well specified with no ambiguity. For the carrying operation, previous task planning provides only the starting position and terminal position for each unit cell. Because one unit cell may have multiple free interfaces, we need to choose appropriate interfaces as P_s and P_t before path planning. For basic unit cells with six identical sides, any free interface can be used as P_s or P_t. But for a unit cell having one side with functional components, such as an antenna, the problem is more complicated. In this work, at a starting position, we choose the free interface as P_s as long as it satisfies the working condition set $\mathcal{W}_c$. For the terminal position, besides the requirements from $\mathcal{W}_c$, we choose the free interface as P_t to facilitate the side of functional component facing the outside, because this side cannot dock with other cells. For the final settlement in $\mathcal{E}$, we select the free interface as P_t that can guarantee to adjust the orientation of the unit cell in case it is specified in expected configuration $\mathcal{E}$.

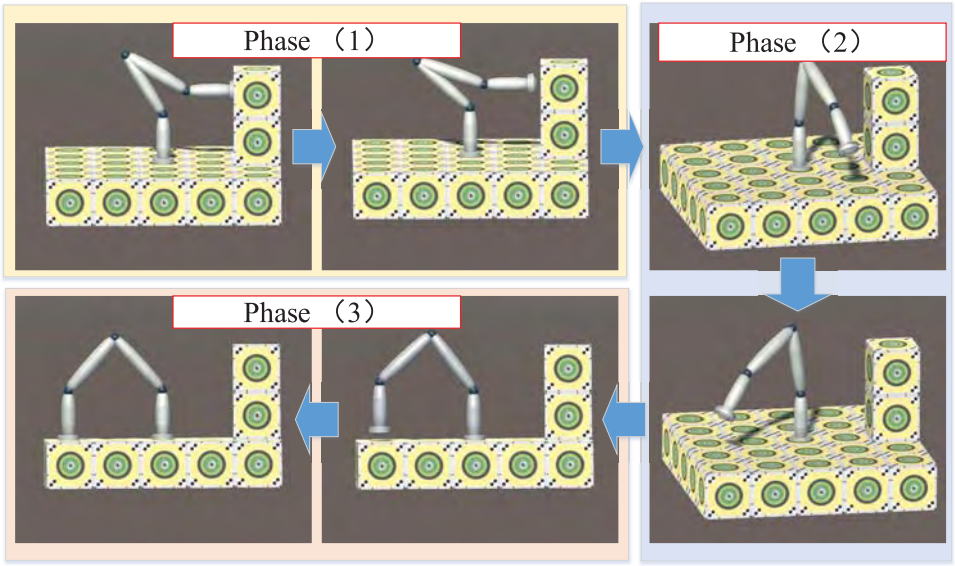


FIGURE 12. Diagram of joint planning.

D. JOINT PLANNING

After path planning, we obtain the foothold locations of the assembling cell. We are now ready to calculate the joint rotations to drive each step motion of the assembling cell. Indeed, joint planning is a standard problem in the robotic arm field. For a self-contained purpose, we briefly introduce the joint planning algorithm used in the proposed strategy.

The inverse kinematics model plays an important role in joint planning. Taking the coordinate system defined in Fig. 4, we define the transformation matrix $\mathbf{T}$ from $\mathcal{F}_0$ to $\mathcal{F}_5$ as [29]

$$\mathbf{T} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{1}$$

We denote the length of the $\mathcal{F}_1$ link of assembling cell as L_1 and the length of the $\mathcal{F}_4$ link as L_4, such that L_1 is the side length of standard unit cell. For crawling and carrying motions, $L_4 = L_1$ and $L_4 = 2\,L_1$ (including a payload cell), respectively. The lengths of other two links are $1.5L_1$. We then follow a regular kinematics derivation process of robotic arm and obtain the inverse kinematics model for the assembling cell [29], [30]

$$\theta_1 = atan(-\frac{p_x + L_2 r_{13}}{p_y + L_2 r_{23}}), \tag{2}$$

$$\theta_2 = atan(a - \frac{2Ma \pm \triangle^{\frac{1}{2}}}{4M}) - atan(\frac{2Ma \pm \triangle^{\frac{1}{2}}}{4M}), \tag{3}$$

$$\theta_3 = atan(\frac{2Ma \pm \triangle^{\frac{1}{2}}}{4M}), \tag{4}$$

$$\theta_4 = atan2(\frac{r_{13}}{sin(\theta_1)}, r_{33}) - atan(\frac{2Ma \pm \triangle^{\frac{1}{2}}}{4M}), \tag{5}$$

$$\theta_5 = atan(-\frac{r_{32}}{r_{31}}), \tag{6}$$

where

$$M = \frac{a^2 + b^2}{3L_1}, \tag{7}$$

$$\triangle = 4M^2a^2 - 4(M^2 - a^2) - 4(M^2 - b^2)(a^2 + b^2), \tag{8}$$

and

$$a = \frac{2}{3L_1}(p_z + L_2 r_{33} - L_1), \tag{9}$$

$$b = \frac{2}{3L_1 cos(\theta_1)}(p_y + L_2 r_{23} c_1). \tag{10}$$

Note that θ_2, θ_3 and θ_4 have multiple solutions and we can choose one solution as needed for each assembling cell motion.

Similar to other studies on robotic arms, once the desired trajectory for the end effector of the robotic arm has been determined, rotational joints can be calculated directly by using the inverse kinematics model. In the following, we take the crawling motion as example to present our considerations on trajectory planning. For one step crawling, we consider the motion of the assembling cell to be composed of three phases (Fig. 12):

1) SEPARATING FROM CELLULAR SATELLITE

The end of the assembling cell vertically leaves the cellular satellite for 0.2 unit distance (the side length of a standard unit cell is 1 unit) to separate the standard interfaces.

2) MOVING END TO DESIRED POSITION

A possible trajectory to the desired position (defined by the starting position in the next phase) is planned and computed for the assembling cell by using a traditional five times polynomial interpolation algorithm. Clearly, the trajectory is not unique. In this work, we randomly sample five trajectories

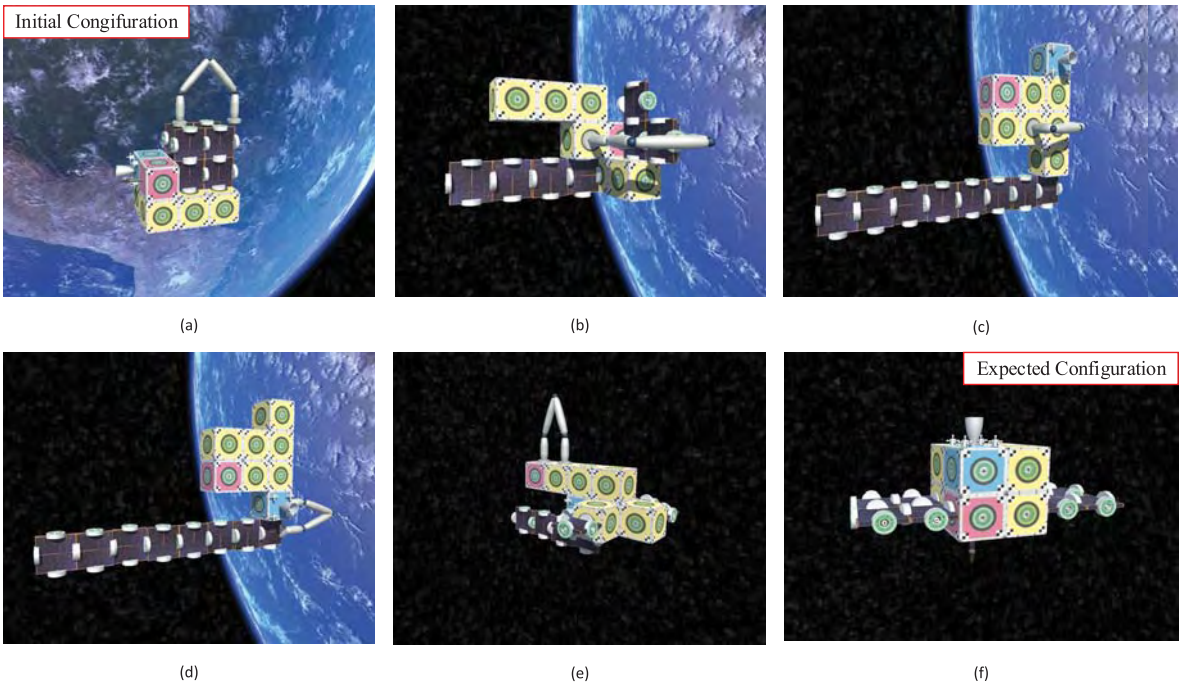


FIGURE 13. Simulation results of Example 1.

and choose one with maximum safe allowance in the environment.

3) DOCKING WITH CELLULAR SATELLITE

The end of the assembling cell approaches the cellular satellite at 0.2 unit distance to dock the interfaces.

We then calculate joint rotational angles by using the inverse kinematics model with the planned trajectory [29], [31]. Similar joint planning results can be obtained for the carrying motion and multiple-cell manipulating motion, and we do not present them here. In the last settling operation for each cell when constructing the expected configuration, the orientation of unit cell will be adjusted if needed by using the assembling cell. This is another consideration in joint planning.

V. SIMULATIONS

In this study, we develop a specific software environment in Unity 3D (Unity Technologies) to validate the proposed self-reconfiguration planning strategy. We built the simulation environment in Unity 3D because it can provide high-fidelity visual simulation effects. In this paper, we present the validation results for three examples.

A. EXAMPLE 1

First, we apply the proposed strategy to a cellular satellite with 16 unit cells. Fig. 13 presents six key snapshots during the self-reconfiguration process. The simulation video can be downloaded from footnote link.[1] The initial and expected configurations are shown in Fig. 13(a) and Fig. 13(f), respectively. Fig. 13(b) presents the melting step from the initial configuration to the intermediate configuration by using the assembling cell. Fig. 13(c) demonstrates the temporary configuration $\mathcal{I}'$ after melting step. Fig. 13(d) shows the similarity intermediate configuration $\mathcal{I}_s$ after the sorting step. Fig. 13(e) illustrates the growing step beginning with the similarity intermediate configuration and the expected configuration is finally constructed in Fig. 13(f). Clearly, the cellular satellite successfully changes its morphology step-by-step from the initial configuration to expected configuration. This example depicts 66 times cell movements and manipulations.

[1] https://pan.baidu.com/s/1VSwWsiQ9BhKBAjtK_AWNgQ

B. EXAMPLE 2

Example 2 is for a cellular satellite with 28 unit cells. Fig. 14 presents six key snapshots during the self-reconfiguration process. Similar to example 1, the simulation video can be downloaded from the link on the footnote. The initial configuration and expected configuration are shown in Fig. 14(a) and Fig. 14(f), respectively. Fig. 14(b) presents the melting step. Fig. 14(c) and Fig. 14(d) show configurations $\mathcal{I}'$ and $\mathcal{I}_s$, respectively. Fig. 14(e) shows the growing step and Fig. 14(f) shows the expected configuration. Again, the cellular satellite autonomously changes its morphology step-by-step by using the assembling cell. For this example, there are totally 138 cell movements and manipulations.

C. EXAMPLE 3

We finally conduct a simulation for 128 unit cells. Similar to the above two example, Fig. 15(a)-(f) present six

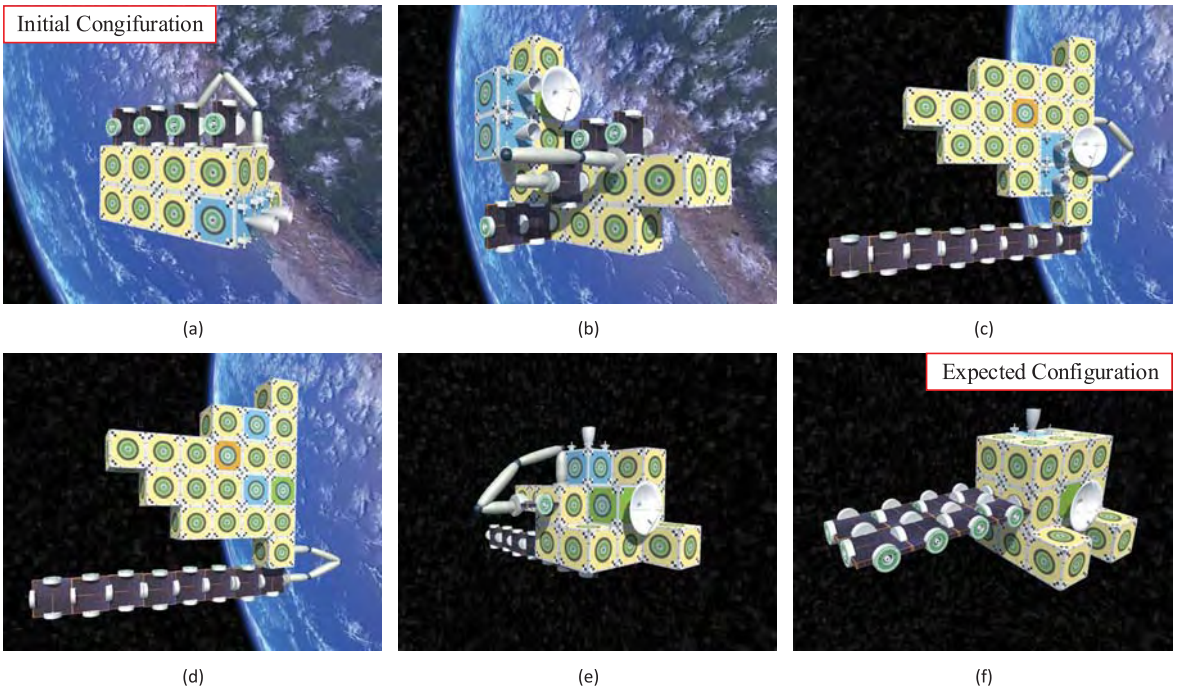


FIGURE 14. Simulation results of Example 2.

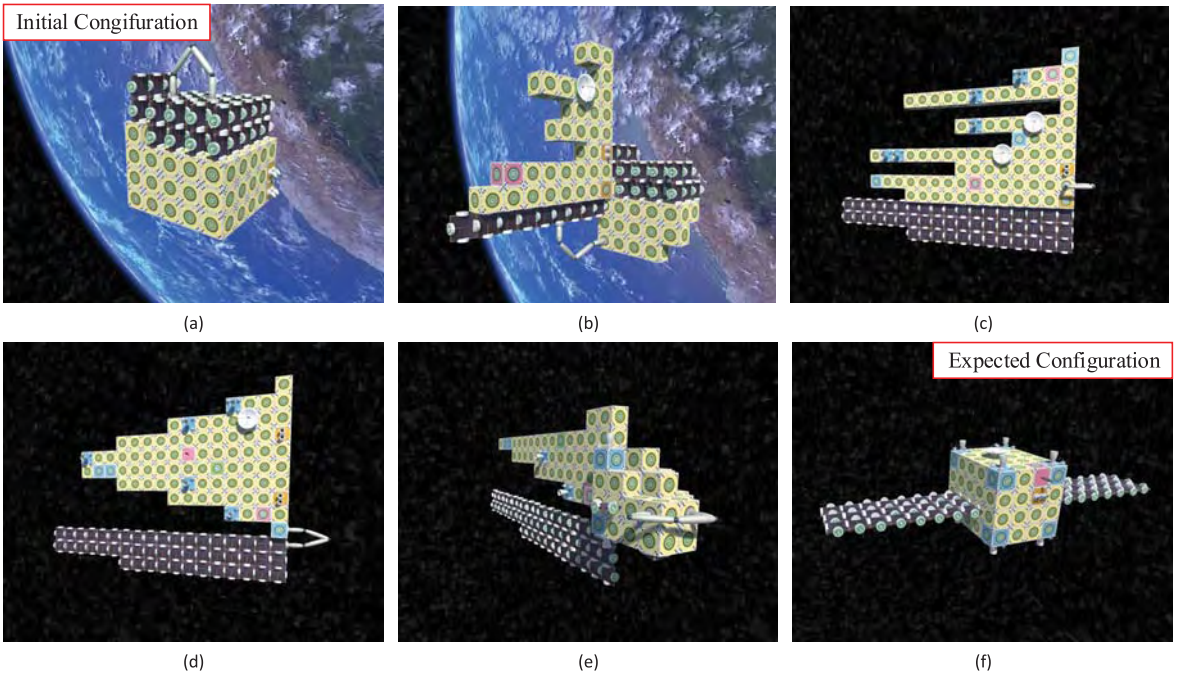


FIGURE 15. Simulation results of Example 3.

key snapshots during the self-reconfiguration process. With assistance from the assembling cell, the cellular satellite successfully changes its morphology from the initial configuration to the expected configuration. In this example, the self-configuration process contains 586 cell movements and manipulations. Clearly, as the number of cells increases, the strategy becomes less efficient because only one assembling cell is used in the strategy, which requires more effort to adjust and sort cell positions. That is one limitation of the strategy with one assembling cell.

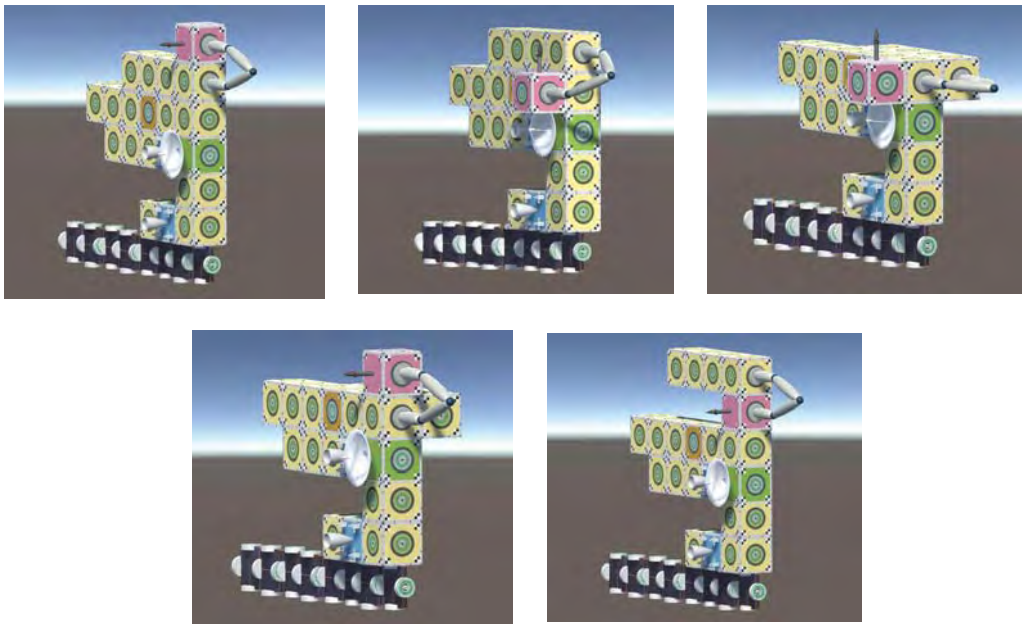


FIGURE 16. One sorting example.

D. SORTING STEP

We present one simulation example of the sorting step to further illustrate Remark 2. As explained previously, we use a similar treatment for the Tower of Hanoi problem. For example, in order to exchange the order of top two $\boldsymbol{l}_i$s in the configuration in Fig. 16, we temporarily attach these two $\boldsymbol{l}_i$s to the side of current configuration to expose the necessary port and then re-arrange their order. Fig. 16 illustrates the sorting step.

VI. CONCLUSIONS

In this paper, we propose a systematic self-reconfiguration planning strategy for cellular satellites. An assembling cell is used to carry or manipulate unit cells to achieve the self-reconfiguration capacity. Corresponding planning algorithms are presented and discussed in this paper. The simulation results verify that by using this strategy, the cellular satellite can autonomously evolve its morphology from an arbitrary initial configuration to an arbitrary expected configuration. We are currently working to further improve the efficiency of this self-reconfiguration strategy by using multiple assembling cells.

REFERENCES

[1] B. L. Benedict, "Rationale for need of in-orbit servicing capabilities for GEO spacecraft," in *Proc. AIAA SPACE Conf. Expo.*, 2013, p. 5444.

[2] D. E. Hastings, B. L. Putbrese, and P. A. La Tour, "When will on-orbit servicing be part of the space enterprise?" *Acta Astronaut.*, vol. 127, pp. 655–666, Nov. 2016.

[3] L. Hill *et al.*, "The market for satellite cellularization: A historical view of the impact of the satlet morphology on the space industry," in *Proc. AIAA SPACE Conf. Expo.*, 2013, p. 5486.

[4] A. Flores-Abad, O. Ma, K. Pham, and S. Ulrich, "A review of space robotics technologies for on-orbit servicing," *Prog. Aerosp. Sci.*, vol. 68, no. 8, pp. 1–26, Jul. 2014.

[5] C. A. Henry, "An introduction to the design of the Cassini spacecraft," *Space Sci. Rev.*, vol. 104, nos. 1–4, pp. 129–153, 2002.

[6] G. H. Fountain *et al.*, "The new horizons spacecraft," *Space Sci. Rev.*, vol. 140, nos. 1–4, pp. 23–47, 2008.

[7] A. Toorian, K. Diaz, and S. Lee, "The CubeSat approach to space access," in *Proc. Aerosp. Conf.*, Mar. 2008, pp. 1–4.

[8] S. Waydo, D. Henry, and M. Campbell, "CubeSat design for LEO-based Earth science missions," in *Proc. Aerosp. Conf.*, Mar. 2002, p. 1.

[9] K. D. Kumar, H. C. Bang, and M.-J. Tahk, "Satellite formation flying using along-track thrust," *Acta Astronaut.*, vol. 61, nos. 7–8, pp. 553–564, 2007.

[10] K. D. Kumar and A. Zou, "Robust stationkeeping and reconfiguration of underactuated spacecraft formations," *Acta Astronaut.*, vol. 105, no. 2, pp. 495–510, 2014.

[11] J. Chu, J. Guo, and E. Gill, "An architecture of on-board autonomy for cluster flight of fractionated spacecraft modules," in *Proc. Int. Astron. Congr.*, 2011, pp. 7109–7119.

[12] J. Chu, J. Guo, and E. Gill, "Decentralized autonomous planning of cluster reconfiguration for fractionated spacecraft," *Acta Astronaut.*, vol. 123, pp. 397–408, Jun. 2016.

[13] D. Barnhart, L. Hill, M. Turnbull, and P. Will, "Changing satellite morphology through cellularization," in *Proc. AIAA SPACE Conf. Expo.*, 2012, p. 5262.

[14] A. A. Kerzhner *et al.*, "Architecting cellularized space systems using model-based design exploration," in *Proc. AIAA SPACE Conf. Expo.*, 2013, p. 5371.

[15] H. Tanaka, N. Yamamoto, T. Yairi, and K. Machida, "Reconfigurable cellular satellites maintained by space robots," *J. Robot. Mechatron.*, vol. 18, no. 3, pp. 356–364, 2006.

[16] M. Goeller, J. Oberlaender, K. Uhl, A. Roennau, and R. Dillmann, "Modular robots for on-orbit satellite servicing," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, Dec. 2012, pp. 2018–2023.

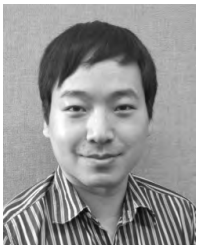
[17] M. Kortmann, A. Dafnis, and H. G. Reimerdes, "Development and breadboard testing of a mechanical coupling interface for modular spacecraft systems," in *Proc. Eur. Conf. Spacecraft Struct., Mater. Environ. Test.*, 2014. [Online]. Available: http://articles.adsabs.harvard.edu/cgi-bin/nph-iarticle_query?2014ESASP.727E..15K&data_type=PDF_HIGH&whole_paper=YES&type=PRINTER&filetype=.pdf

[18] *Phoenix Program Status 2013*, AIAA, Reston, VA, USA, 2000.

[19] P. Melroy *et al.*, "DARPA phoenix satlets: Progress towards satellite cellularization," in *Proc. AIAA SPACE Conf. Expo.*, 2015, p. 4487.

[20] H. Chang, P. Huang, Y. Zhang, Z. Meng, and Z. Liu, "Distributed control allocation for spacecraft attitude takeover control via cellular space robot," *J. Guid., Control, Dyn.*, vol. 41, no. 11, pp. 2499–2506, 2018.

[21] H. Chang, P. Huang, Z. Lu, Y. Zhang, Z. Meng, and Z. Liu, "Inertia parameters identification for cellular space robot through interaction," *Aerosp. Sci. Technol.*, vol. 71, pp. 464–474, Dec. 2017.
[22] M. Yim *et al.*, "Modular self-reconfigurable robot systems [grand challenges of robotics]," *IEEE Robot. Automat. Mag.*, vol. 14, no. 1, pp. 43–52, Mar. 2007. [Online]. Available: https://ieeexplore.ieee.org/document/4141032, doi: 10.1109/MRA.2007.339623.
[23] S. Murata and H. Kurokawa, "Self-reconfigurable robots," *IEEE Robot. Autom. Mag.*, vol. 14, no. 1, pp. 71–78, Mar. 2007.
[24] Y. Zhu, D. Bie, S. Iqbal, X. Wang, Y. Gao, and J. Zhao, "A simplified approach to realize cellular automata for ubot modular self-reconfigurable robots," *J. Intell. Robot. Syst.*, vol. 79, no. 1, pp. 37–54, 2015.
[25] Y. Zhu, D. Bie, X. Wang, Y. Zhang, H. Jin, and J. Zhao, "A distributed and parallel control mechanism for self-reconfiguration of modular robots using L-systems and cellular automata," *J. Parallel Distrib. Comput.*, vol. 102, pp. 80–90, Apr. 2017.
[26] W.-M. Shen, B. Salemi, and P. Will, "Hormone-inspired adaptive communication and distributed control for CONRO self-reconfigurable robots," *IEEE Trans. Robot. Autom.*, vol. 18, no. 5, pp. 700–712, Oct. 2002.
[27] R. Fitch, Z. Butler, and D. Rus, "Reconfiguration planning among obstacles for heterogeneous self-reconfiguring robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, Apr. 2006, pp. 117–124.
[28] A. Soltani, H. Tawfik, J. Y. Goulermas, and T. Fernando, "Path planning in construction sites: Performance evaluation of the Dijkstra, A*, and GA search algorithms," *Adv. Eng. Inform.*, vol. 16, no. 4, pp. 291–303, 2002.
[29] J. J. Craig, *Introduction to Robotics: Mechanics and Control*. Reading, MA, USA: Addison-Wesley, 2005.
[30] G. Dong and Z. H. Zhu, "Position-based visual servo control of autonomous robotic manipulators," *Acta Astronaut.*, vol. 115, pp. 291–302, Oct. 2015.
[31] H. Wang, D. Guo, H. Xu, W. Chen, T. Liu, and K. K. Leang, "Eye-in-hand tracking control of a free-floating space manipulator," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 53, no. 4, pp. 1855–1865, Aug. 2017.

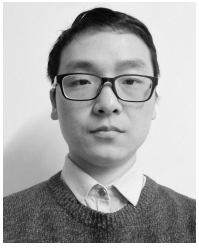


YIZHAI ZHANG (S'11–M'14) received the B.S. and M.S. degrees in information and communication engineering from Xi'an Jiaotong University, China, in 2005 and 2009, respectively, and the Ph.D. degree in mechanical and aerospace engineering from Rutgers University, USA, in 2014. He is currently an Associate Professor with the School of Astronautics, Northwestern Polytechnical University, China. His current research interests include autonomous robotic systems, dynamic systems and control, and mechatronics.

Dr. Zhang is a member of the IEEE and ASME. He received the Best Student Paper Award at the 2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics and the 2012 ASME Dynamic Systems and Control Conference. He was a recipient of the 2013 Chinese Outstanding Overseas Student.

WENHUI WANG received the B.S. degree in aircraft design engineering from Northwestern Polytechnical University, China, in 2017, where she is currently pursuing the master's degree with the Research Center for Intelligent Robotics.

JINGHAN SUN received the B.S. degree in aerospace engineering from Northwestern Polytechnical University, China, in 2017, where he is currently pursuing the master's degree with the Aerospace Department.



HAITAO CHANG was born in Handan, Hebei, China, in 1987. He received the B.S. degree in detection, guidance and control techniques and the M.S. and Ph.D. degrees in navigation, guidance and control from Northwestern Polytechnical University, Xi'an, China, in 2010, 2013, and 2018, respectively. He is currently a Research Associate with the School of Astronautics, Northwestern Polytechnical University.

His research interests include dynamics modeling and distributed cooperative control of cellular space robot, self-reconfiguration strategies of cellular robots, and spacecraft attitude takeover control.

PANFENG HUANG (S'01–M'05–SM'17) received the B.S. and M.S. degrees in test and measurement technology, and navigation guidance and control from Northwestern Polytechnical University, Xi'an, China, in 1998 and 2001, respectively, and the Ph.D. degree in automation and robotics from The Chinese University of Hong Kong, Hong Kong, in 2005.

He is currently a Professor with the School of Astronautics and the National Key Laboratory of Aerospace Flight Dynamics, Northwestern Polytechnical University, where he is also the Vice Director of the Research Center for Intelligent Robotics. His research interests include tethered space robotics, intelligent control, machine vision, and space teleoperation.