Proceedings of the International Conference of Control, Dynamic Systems, and Robotics
Ottawa, Ontario, Canada, May 15-16 2014
Paper No. 116

Multi-robots Cooperative Assembly Planning of Large Space Truss Structures

Jifeng Guo[1,2], Chunlin Song, Cheng Chen
[1]Harbin Institute of Technology
92 West Da Zhi Street, Harbin, China 150001
guojifeng@hit.edu.cn; sclin1217@163.com; wddaqjjxs@163.com

Jinjun Shan
[2]York University, Department of Earth and Space Science and Engineering
4700 Keele Street, Toronto, Ontario, Canada M3J 1P3
jjshan@yorku.ca

Abstract – The large space truss structures will be likely to require on-orbit assembly in the future. One of the several proposed methods include cooperative assembly performed by multi-robots. An intelligent planning method was presented to generate optimal assembly tasks. Firstly, the inherent hierarchical nature of truss structures allows assembly sequences to be considered from strut level and structural volume element (SVE) level. Then, a serial assembly strategy in multi-robots environment was applied. Furthermore, an assembly sequence planning algorithm was presented. At the assembly sequence planning level, one ant colony algorithm for assembly sequence planning was improved to adopt assembly direction and time as heuristic information and considered assembly tasks. And, at the assembly motion planning level, a randomization-based motion planning algorithm for assembly task planning mainly considered results of the mission-level planning, multi-robots interactive information, serial assembly strategy and assembly task distributions. A case illustrates results of planning methods.

***Keywords*:** Randomization, assembly planning, on orbit assembly, large space structures.

1. Introduction

Many future space exploration missions must be supported on large space truss structures and are larger in diameter than any current or proposed launch vehicles. Moreover, because of its size, the space truss structure is likely to require on-orbit assembly. One of the several proposed methods include cooperative assembly performed by multiple space robots. On-orbit assembly technology has the potential to enable a wide range of revolutionary new capabilities for space exploration and development as shown by L. S. Homem-de-Mello. (1995). On-orbit assembly performed by multiple space robots will provide an effective approach to build large space truss structures.

Because of the size and complexity of large space truss structures and multi-robots interactive environment, on-orbit assembly requires careful planning. Assembly processes need to systematize and computerize the generation of the assembly sequences and motion paths in order to guarantee that the components are assembly in a correct, complete and efficient sequence and enable the fast generation of safe motion paths.

Artificial intelligence methods have been developed to solve the assembly sequence planning problem. Ant colony system algorithm was firstly applied to the traveling salesman problem (TSP) by Dorigo. In assembly sequence planning problem, researchers firstly used ant systems to sequence detection and optimization as shown by Failli and Dini. (2000). and a novel ant colony algorithm was presented to generate and optimize assembly sequences of mechanical products as shown by J. F. Wang. (2004).

The motion planning of multiple space robots is inspired mainly in two major directions. A lot of literatures have studied on motion planning of continuous control mode as shown by Ian Garcia, Jonathan P. How. (2005). in recent years. In the other direction, many people also obtained some result. For example, an artificial potential functions was used to generate safe trajectories around space station to dock as shown by Colin R. McInnes. (2000). This kind of problem was solved based on a transcription of the motion planning problem into a Mixed-Integer Linear Program (MILP) as shown by T. Schouwenaars et al. (2001).

Rapidly-exploring Random Trees (RRTs) algorithm based on randomized motion planning not only adapts higher dimensional space planning problem but also considers differential constraints in the generation process of the search trees as shown by S. M. LaValle. (2004). And this planner can effectively avoid existence of local minima. Moreover, RRT planner shows good performance in fast and uniform exploration of the configuration space. The Quasi-Random Algorithms was used based on RRT to solve real-time servicing spacecraft motion planning and coordination in close proximity of one another as shown by Emilio Frazzoli. (2003).

In this paper an intelligent planning method was presented to generate optimal assembly missions. Firstly, the inherent hierarchical nature of truss structures allows assembly sequences to be considered from strut level and structural volume element (SVE) level. At the strut level assembly sequences were represented by the connectivity matrix. And, at the SVE level the directed graph representation was used. Then, a serial assembly strategy in multi-robots environment was applied. Furthermore, a two-levels planning algorithm was presented. At the mission sequence planning level, an ant colony algorithm for assembly sequence planning was improved to adopt assembly direction and time as heuristic information and considered assembly tasks. And, at the robot motion planning level, a randomization-based motion planning algorithm for assembly motion planning mainly considered results of the first-level planning, multi-robots interactive information, serial assembly strategy and assembly task distributions. A case illustrates results of two-level planning. At the first-level planning, the optimal assembly sequence with assembly task conditions was got and respectively included sequences of the strut level and SVE level. At the second-level planning, the results of assembly path planning that considered results of the first-level planning, multiple space robots interactive information, serial assembly strategy and assembly task distributions were given. Finally, the conclusions of multi-Robots cooperative assembly planning of large space truss structures were also got.

2. Representation of Assembly Sequence

The inherent hierarchical nature of the space-truss domain allows 3-D space structures to be views from two levels: struts and nodes interconnected to form a complete super-structure and collections of struts and nodes that form rigid sub-structures that are interconnected to form the complete structure. The representations of large space truss structures are respectively presented at the SVE level and strut level.

A structural volume element (SVE) is a collection of struts and nodes that, when interconnected in a specific manner, forms a structurally rigid and stable state. SVE elements can be pentahedron (or pyramid), tetrahedron, braced cubes, or any other rigid, 3-D structure that can be used as a building block or substructure to build a larger, more expansive 3-D structure.

At the strut level, assembly units are every struts of SVE. The structural volume element is represented by the connectivity matrix at this level. The connectivity matrix is defined by

$$\varepsilon(i,j) = \begin{cases} 1 & , Connection\ between\ r_i\ and\ r_j \\ 0 & , No\ connection\ between\ r_i\ and\ r_j \end{cases} \quad (1)$$

Where r_i and r_j correspond to the *ith* and *jth* struts and (i,j) corresponds to the matrix element of column i and row j. The dimensions of $\varepsilon(i,j)$ are same to the total number of SVE struts. The assembly process starts from the top left corner of ε_{ij}.

3. Assembly Mission Sequence Planning Level

Two-levels hierachical planning considers assembly planning at the mission sequence planning level and robot motion planning level. At the mission sequence planning level, an ant colony algorithm for assembly sequence planning was improved to adopt assembly direction and time as heuristic information and considered assembly tasks.

In this paper, the algorithm is based on the following assumptions: a) the SVE is the basic component; b) disassembly sequences of components inversely represent the assembly sequences; c) assembly sequence quality is evaluated by assembly time.

During the process of the first-level planning, an improved ant colony algorithm is presented and considered results of the first-level planning, multi-robots interactive information, serial assembly strategy and assembly task distributions.

The state-transition probability means the evaluation with which ant *k* moves from the SVE *i* to the SVE *j* at the time *t* and is defined by

$$Q_{ij}^{k} = \begin{cases} \dfrac{[\sigma_{ij}(t)]^{a}[\omega_{ij}(t)]^{b}[S_{ij}(t)]^{c}}{\sum_{k\in allowed_k}[\sigma_{ik}(t)]^{a}[\sigma_{ik}(t)]^{a}[\sigma_{ik}(t)]^{a}} & ,if\ j \in allowed_k \\ 0 & ,otherwise \end{cases} \tag{2}$$

where $\sigma_{ij}(t)$ is the amount of pheromone on sequence (i,j) at the time t; $\omega_{ij}(t)$ is the heuristic information of moving from the SVE i to the SVE j, corresponding to mission information of space robots in sequence (i,j), and is calculated as

$$\omega_{ij}(t) = \begin{cases} 1 & ,if\ no\ mission \\ 0.1 & ,otherwise \end{cases} \tag{3}$$

$S_{ij}(t)$ is another heuristic information from the SVE *i* to the SVE *j*, corresponding to the change of numbers of astronauts that is needed in sequence (*i*,*j*),

$$S_{ij}(t) = \begin{cases} 1 & ,if\ decreased \\ 0.5 & if\ don't\ changed \\ 0.1 & ,if\ increased \end{cases} \tag{4}$$

$allowed_k$ is the set of allowed neighbors of the SVE *i*; the exponents *a*, *b* and *c* are used to determine the relative importance of $\sigma_{ij}(t)$, $\omega_{ij}(t)$ and $S_{ij}(t)$.

While all ants have completed searching, pheromone evaporation on all sequences is triggered, and then each ant *k* deposits a quantity of pheromone $\Delta\sigma_{ij}^{k}(t)$ on each sequence that it can be represented as:

$$\Delta\sigma_{ij}^{k}(t) = \begin{cases} \dfrac{W}{L_k(t)} & ,if\ sequence(i,j) \in sequence \\ 0 & ,otherwise \end{cases} \tag{5}$$

where *W* is a constant and $L_k(t)$ corresponds to the assembly sequence quality.

The quantity of pheromone $\Delta\sigma_{ij}(t)$ on the whole sequences is calculated as

$$\Delta\sigma_{ij}(t) = \sum_{k=1}^{m}\Delta\sigma_{ij}^{k}(t) \tag{6}$$

$\rho(t)$ is the pheromone trail decay coefficient.

Because of pheromone evaporation, the addition of new pheromone is implemented by the following rule

$$\sigma_{ij}(t+\Delta t) = (1-\rho)\cdot\sigma_{ij}(t) + \Delta\sigma_{ij}(t) \tag{7}$$

4. Multi-robots Assembly Motion Planning Level

In this Section, multi-robots assembly motion planning algorithm is described in detail. It belongs to motion and constraints of the space robots, the formulation of the algorithm and schemes.

4. 1. Problem Description

The explicit equation of space robots relative motion is

$$\begin{bmatrix} r_t \\ v_t \end{bmatrix} = \emptyset \begin{bmatrix} r_0 \\ v_0 \end{bmatrix} = \begin{bmatrix} \emptyset_{11} & \emptyset_{12} \\ \emptyset_{21} & \emptyset_{22} \end{bmatrix} \begin{bmatrix} r_0 \\ v_0 \end{bmatrix} \tag{8}$$

Space robots move to the final state from initial state according to *N* velocity impulses with impulse thrusters. Using the explicit equation (8) of relative motion, the ΔV_i^- and ΔV_i^+ for i=1...N velocity impulse vectors, are

$$\Delta V_i^- = \emptyset_{12}^{-1}(t_i+\Delta t)[r(t_i+\Delta t) - \emptyset_{11}(t_i+\Delta t)r(t_i)] \tag{9}$$

$$\Delta V_i^+ = \emptyset_{21}(t_i+\Delta t)r(t_i) + \emptyset_{22}(t_i+\Delta t)v(t_i) \tag{10}$$

Where $r(t_i)$and $r(t_i+\Delta t)$ are respective initial and final position vector in the *ith* impulse motion. $v(t_i)$ is initial velocity vector and equal to ΔV_i^-. t_i is initial time in the *ith* impulse motion.

Therefore, velocity increment vector is

$$\Delta u_i = \Delta V_i^+ - \Delta V_i^- \tag{11}$$

Then the first kind of constraint we consider encodes ΔV, and takes the form

$$0 \le \Delta V_i^+, \Delta V_i^- \le \Delta V_{max} \tag{12}$$

Collision avoidance is encoded by constraints of the form

$$\mathrm{Obs}[r(t), t] < 0, \forall t \tag{13}$$

Finally, plume impingement constraints are encoded by

$$\mathrm{Plu}[r(t), \Delta V, t] < 0, \forall t \tag{14}$$

The class of problem considered in this paper can be formulated in terms of below components:

1) State Space: A topological space and its obstacles are X and $X_{obs}(X_{obs} \in X)$, the violation-free set is $X_{free}\left(X_{free} = X/X_{obs}\right)$.

2) Boundary Values: $X_{init} \in X_{free}$ and $X_{goal} \in X_{free}$.

3) Constraints: Constraints (12~14) are satisfied for servicing spacecraft at all times.

4) Incremental Simulator: Give the current state *x(t)* and inputs (ΔV impulse) applied at current time *t* and time interval Δt, compute $x(t+\Delta t)$.

5) Metric: A real-valued function ρ, which specifies the distance between pairs of points in X_{free}.

4. 2. Randomized Motion Planning Algorithm

The motion planning algorithm for an arbitrary elliptical orbit similar to general randomization-based motion planning like [6] consists of branches, milestones and trees composed by them. These trees also start from initial conditions grow to the goal set, and become a feasible trajectory with respect to the constraints. But there are some differences in this algorithm because of characteristics of elliptical orbit. The one of most important of these differences is the simulation time. The dynamical environment on an elliptical orbit is constantly changing with time. This point is found in the explicit equation (8). Its true anomaly as independent variable is equivalent to the time in fact. Therefore, the problem of time is emphasized in this planner.

Table 1. Randomized motion planning algorithm.

Motion_Planning(x_{init}, x_{end})
1 $Tra = \emptyset, Seq\Delta V = \emptyset, Time = \emptyset, V = \{x_{init}\}$
2 $Tree_{init} = \{x_{init}\}, seq\Delta V_{init} = \emptyset, seqTime_{init} = \{0\}$
3 $For\ i = 1\ to\ I$
4 Build_Tree(x_{init}, x_{end})
5 $Tra = Tra \cup tra_i$
6 $Seq\Delta V = Seq\Delta V \cup seq\Delta V_i$
7 $SeqTime = SeqTime \cup seqTime_i$
8 *Return* Tra

The basic randomized motion planning algorithm in this paper is given in Table. 1. *Tra* is a set of solutions of the trajectories composed by many way points. *Seq*ΔV is a sequence of ΔV consist of ΔV^- and ΔV^+ for impulse. *SeqTime* is a sequence of initial times as independent variable corresponding to the every way points. V is a set of all milestones. tra_i, $seq\Delta V_i$ and $seqTime_i$ are the solution of the trajectory, sequence of ΔV and sequence of initial time for the *ith* milestone, respectively. $Tree_i$, starts from (x_0, t_0) and ends to current milestone, is a tree of the *ith* milestone.

This algorithm can obtain I solutions of trajectories expressed in the 3rd line in Table. 1. Build_Tree() is the main function used to acquire every of them in the 4th line. The first object of this function is calculating the first and second milestone for expanding the tree (see Table. 2). If target point is reachable from the milestone (1st or 2nd) obtained (i.e. a feasible trajectory is found), $x_{fms}$and x_{sms} were added in the set of milestones V (see 4th line in Table. 2).

Table 2. The Build_Tree() function

Build_Tree(x_{init}, x_{end})
1 While Constraints$(x_{fms}, x_{end}) \geq 0\ \ and\ \$ Constraints$(x_{sms}, x_{end}) \geq 0$
2 First_Milestone()
3 Second_Milestone($x_{fms}, \Delta V^+_{fms}, t_{fms}$)
4 $tra_i = Tree_{ms} \cup \{x_{end}\}$
5 $seq\Delta V_i = seq\Delta V_{ms} \cup \{\Delta V_{end}\}$
6 $seqTime_i = seqTime_{ms} \cup \{Time_{end}\}$

The basic steps of first milestone are the following in Table. 3:

① Sample uniformly a candidate milestone x_{rand} in x_{free}.

② Make a set of neighbours of x_{rand} chosen from V named N_n ($N_n = V$ in general), n_n is the number of neighbours.
③ Sample uniformly a flying time $t_p(p = 1 \cdots N_s)$ which starts from parent node and ends to candidate milestone x_{rand}. N_s is the number of sampling ($N_s = 256$ in general). Obtain $\Delta V_p^+$and ΔV_p^- from relative motion explicit equation (11) and (12) for every neighbour $x_j\left(x_j \in N_n\right)$. If ΔV_p^+, ΔV_p^- and the edge of x_j to x_{rand} satisfy the constraints (10~12), x_j is saved to the set of parent nodes V_{pn}. Otherwise, the new candidate milestone x_{rand} is sampled. Obtain t_j, the minimum of t_p, and $\Delta V_{t_j}^+$, $\Delta V_{t_j}^-$ relevant to t_j.
④ If V_{pn} is not empty, find the nearest neighbour x_{pn}(parent node) from current x_{rand} by the cost *J*.

$$J = \min\left[\sum_{i=1}^{N}(\|\Delta V_i^+\|_1 + \|\Delta V_i^-\|_1) + kT\right] \tag{15}$$

Where *N* is the number of impulses, *T* is the total of the flying time, and *k* is a weight that relatives ΔV and *T*.
⑤ Obtain new first milestone x_{fms}, and its ΔV_{fms}, t_{fms}, $Tree_{ms}$, $seq\Delta V_{ms}$ and $seqTime_{ms}$. t_{fms} is the flying time that starts from initial node to the new first milestone as independent variable to calculate the next first milestone used by explicit equation.

Table 3. The First_Milestone() function.

First_Milestone( )	
1	$V_{pn} = \emptyset$
2	While $V_{pn} = \emptyset$
3	$x_{rand} \leftarrow$ RandomState()
4	$N_n = a\ set\ of\ neighbors\ of\ x_{rand}\ chosen\ fron\ V$
5	For j=1 to n_n
6	For $p \leftarrow 1\ to\ N_s$
7	$t_p \leftarrow$ RandomTime(T_{max})
8	$\Delta V_p^+, \Delta V_p^- \leftarrow$ Explicit_Equation(x_j, x_{rand}, t_p, t_j)
9	If Constraints(x_j, x_{rand}) < 0
10	$V_{pn} = V_{pn} \cup \{x_j\}$
11	$t_j \leftarrow \min\{t_1, t_2, \cdots, t_p\}$
12	$\Delta V^+ \leftarrow \Delta V_{t_j}^+,\ \Delta V^- \leftarrow \Delta V_{t_j}^-$
13	$x_{pn} \leftarrow$ Nearest_Neighbor(V_{pn}, x_{rand})
14	$x_{fms} \leftarrow x_{rand},\ \Delta V_{fms} = \{\Delta V^+,\ \Delta V^-\},\ t_{fms} \leftarrow t_{pn} + t_j$)
15	$Tree_{ms} = Tree_{pn} \cup \{x_{fms}\}$
16	$seq\Delta V_{ms} = seq\Delta V_{pn} \cup \Delta V_{fms}$
17	$seqTime_{ms} = seqTime_{pn} \cup \{t_{fms}\}$

The second milestones are added to the tree (i.e. an edge which starts from parent node to the first milestone is split into two edges, the second milestone is inserted in between). Applying of the second milestone can efficiently increase the probability of obtaining solution. Its method looks like the first milestone expressed in Table. 4. n_s is the number of the second milestones expected. Similarly, the t_{rand} is flying time between t_{pn} and t_{fms}.

Table 4. The Second_Milestone() function.

Second_Milestone($x_{fms},\ \Delta V_{fms}^{+},\ t_{fms}$)	
1	For $j = 1\ to\ n_s$
2	$t_{rand} \leftarrow \text{RandomTime}(t_{fms} - t_{pn})$
3	$(x_{sms}, \Delta V_{sms}^{-}) \leftarrow \text{Explicit\_Equation}(x_{pn}, \Delta V_{fms}^{+}, t_{rand}, t_{pn})$
4	$\Delta V_{sms} = \{\Delta V_{fms}^{+},\ \Delta V_{sms}^{-}\},\ t_{fms} \leftarrow t_{pn} + t_{rand}$
5	$Tree_{ms} = Tree_{pn} \cup \{x_{sms}\}$
6	$seq\Delta V_{ms} = seq\Delta V_{pn} \cup \Delta V_{sms}$
7	$seqTime_{ms} = seqTime_{pn} \cup \{t_{sms}\}$

5. Examples

In this paper, the example is the structure shown in Fig. 1 and consists of 10 nodes, 24 struts and 7 SVE elements that include three pentahedron and four tetrahedron. In (b) of Fig. 1, the rhombus corresponds to the pentahedron and the triangle corresponds to the tetrahedron.

In this article, the case study is based on the following assumptions: each tetrahedron is assembled by two astronauts; each pentahedron is assembled by three astronauts; the assembly directions are defined by polar coordinates directions that include clockwise (+) and counter-clockwise (-) directions.

The parameters of ACO algorithm have been chosen as: m=15, t=0, $N_{cmax} = 100$, a=1, b=0.8, c=0.1, W=10, $\rho = 0.1$ and $\sigma_0 = 1$.

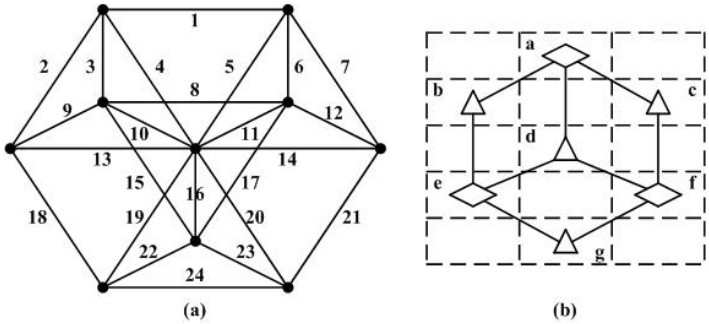

Fig. 1. Case study: (a) the structure; (b) the directed graph representation.

Table 5. Results of the Assembly Mission Sequence Planning.

Level		Mission Sequence
Beam Level	c	(11,1), (5,2), (7,3), (6,2), (12,1), (20,3)
	f	(17,2), (21,3), (14,3), (16,2), (23,1)
	g	(24,3), (22,1), (13,2)
	e	(18,2), (15,1), (10,1), (19,2), (9,3)
	b	(4,1), (2,2), (3,3)
	a	(8,3), (1,1)

The results of assembly mission sequence planning are shown in Table. 5. From this case study, because of geometrical symmetry nature of large space truss structures, the optimal assembly sequences aren't only one.

Space robots manoeuvrer around static spherical obstacle considered in this section and avoids collisions and plume impingement. 200 first milestones and 200 second milestones are obtained, and 194 feasible trajectories are found. The results of multi-robots motion planning are shown in Fig. 2.

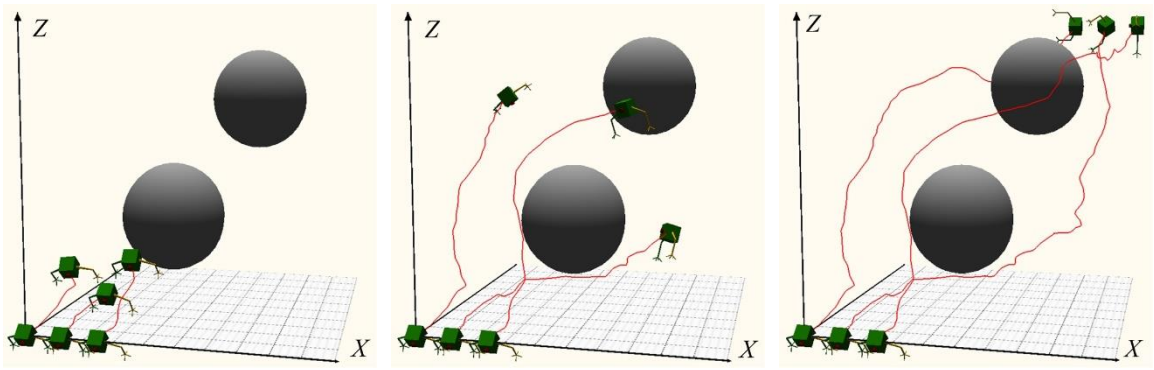

Fig. 2. The Results of Multiple Space Robots Motion Planning.

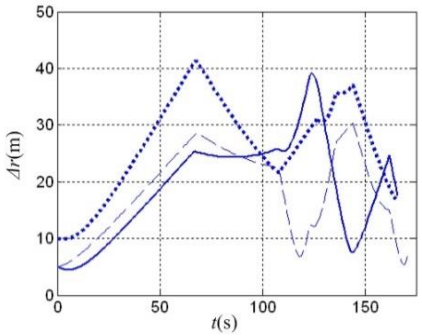

Fig. 3. The Relative Distance between Three Trajectories.

The Relative Distance between Three Trajectories are shown in Fig. 3 and the costs of three feasible trajectories are shown in Table. 6.

Table 6. The Costs of Three Feasible Trajectories.

Flying Time (s)	ΔV(m/s)	J
193.7	0.7916	0.32929
228.8	0.33124	0.33124
195.4	0.39134	0.72352

Table 7. Results of the Simulation with Different Steps.

Δt(s)	t(s)	Feasible Results			Solution Time(s)
0.05	546	221	350	323	342
0.1	437	238	241	272	161
0.5	353	143	172	165	32
1	341	79	95	172	19
2	337	34	73	130	12

The Δt of simulation step have been chosen as 0.05, 0.1, 0.5, 1 and 2. The results of the simulation with different steps are shown in Table. 7.

So in this case, two-phase planner is represented and proven to adapt for the conditions of larger range of motion and more complicated obstacles.

4. Conclusion

In conclusion, this paper deals with the application of the intelligent planning method to the problem of cooperative assembly planning in multi-robots environment. At the first-level planning, one ant colony algorithm for assembly sequence planning was improved to adopt assembly direction and time as heuristic information and didn't considered assembly tasks. And, at the second-level planning, a randomization-based planning algorithm for assembly motion planning. From case studies, the feasibility and validity of the proposed approach are demonstrated. Finally, the two-level planning algorithm can deal with the assembly sequence and motion planning in multiple space robots environment more effectively.

Acknowledgements

This work is partially supported by China Civil Aerospace Advance Research Project Funding, China National Nature Science Funding (61005060) and Key Laboratory Opening Funding of Technology of Micro-Spacecraft.

References

Chen, G., Zhang L., Jia Q, Chu M., Sun H. (2013). Repetitive motion planning of free-floating space manipulators. International Journal of Advanced Robotic Systems, 10, 1–11.

Failli, F., Dini. G. (2000). Ant colony systems in assembly planning: a new approach to sequence detection and optimization, "Proceedings of the 2nd CIRP International Seminar on Intelligent Computation in Manufacturing Engineering", 227-232.

Frazzoli E (2003). Quasi-random algorithms for real-time spacecraft motion planning and coordination, Acta Astronautic, 485-494.

Garcia I., How J.P. (2005). Trajectory optimization for satellite reconfiguration maneuvers with position and attitude constraints, "American Control Conference. Portland, OR, USA", vol. 2, 889-894.

Homem-de-Mello, L. S. (1995). Sequence planning for robotic assembly of tetrahedral truss structures, IEEE Transactions on Systems, Man, and Cybernetics, 25, 304-312.

Larouche B.P., Zhu G.Z.H (2013). Investigation of impedance controller for autonomous on-orbit servicing robot. Canadian Aeronautics and Space Journal, 59(1), 15–24.

LaValle S. M. (2004). "Planning algorithms," Website: http://msl.cs.uiuc.edu/ planning/.

Roger, A. B., McInnes C. R. (2000). Safety constrained free flyer path planning at the international space station, AIAA J. on Guidance, Control an Dynamics, 23(6), 971-979.

Schouwenaars, T., De Moor, B., Feron, E., How, J. (2001). Mixed integer programming for multi-vehicle path planning, "European Control Conference. Porto, Portugal", 2603-2608.

Wang, J. F., Liu J. H., Zhong Y. F. (2004). A novel ant colony algorithm for assembly sequence planning, International Journal of Advanced Manufacturing Technology, 25, 1137-1143.