2022 19th International Conference on Ubiquitous Robots (UR)
July 4-6, 2022. Ramada Plaza Jeju, Jeju, Korea

2022 19th International Conference on Ubiquitous Robots (UR) | 978-1-6654-8253-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/UR55393.2022.9826275

Simulation Study on the Locomotion Algorithm of Variable Topology Truss Robot based on Motion Primitives

Yecheol Moon[1], Jangho Bae[2], Mark Yim[2] and TaeWon Seo[1], *Senior Member, IEEE*

***Abstract*— Variable Topology Truss (VTT) is a modular robotic system, which is an extended robot concept based on the Variable Geometry Truss (VGT). It is composed of modular members and nodes and can operate in a rough and cluttered environment. By changing the length of each member, the VTT robot can change size and shape. The VTT can also perform locomotion by rolling. Travel over a long distance can be accomplished by repeating the rolling motion, so a motion primitive is used. This reduces the computational work because the repetitive motion only needs to be calculated once. In this paper, an algorithm is presented to derive a motion primitive for an octahedral VTT. The algorithm consists of three phases: placing one node on the ground, moving the center of mass, and returning to the initial shape. On each time step, a set of possible robot motions is evaluated based on a cost function. The lowest cost motion is selected on each time step to form the optimized motion primitive. The resulting motion primitive is analyzed and validated.**

I. INTRODUCTION

The Variable Geometry Truss (VGT) is a truss structure that can expand and contract through member length change. VGT was first introduced by Miura [1]. VGT was initially studied mainly as a modification of the structure. Miura et al. introduced a structure capable of changing the length of the crane arm of the space [2]. Hughes et al. introduced a manipulator arm using a VGT called Trussarm [3]. In the case of rolling locomotion for modular robots, the method of locomotion by moving the center of mass is mainly used [4]. Lee and Sanderson proposed a method of rolling locomotion for VGTs by considering two phases and switching the control sequence [5].

Variable Topology Truss (VTT) is a modular robot system based on VGTs that can freely change its topology and size. It is designed to be used in rough terrain to take advantage of VTT's changeable characteristics. To implement the concept of VTT, reconfigurable hardware modules have been proposed [6]. The reconfiguration capabilities of the VTT have been described using a graph-based methods [7] and motion planning algorithms [8]. Trajectory generation based on the random tree search algorithm was studied [9], [10].

This research was supported by the MOTIE (Ministry of Trade, Industry, and Energy) in Korea, under the Human Resource Development Program for Industrial Innovation(Global) (P0017306, Global Human Resource Development for Innovative Design in Robot and Engineering) supervised by the Korea Institute for Advancement of Technology (KIAT).

[1]Yecheol Moon and TaeWon Seo are with the Department of Mechanical Convergence Engineering, Hanyang University, Seoul 04763, Republic of Korea. mycm1302@hanyang.ac.kr; taewonseo@hanyang.ac.kr

[2]Jangho Bae and Mark Yim are with Mechanical Engineering and Applied Mechanics, University of Pennsylvania, Philadelphia, PA 19104, USA jangho.bae91@gmail.com; yim@seas.upenn.edu

To reduce the computational cost in trajectory planning, studies based on motion primitives have been conducted. This method is commonly applied to other robots. Hauser et al. introduce a new step motion primitive for the humanoid robot walking on varied terrain by using a sample-based planner [11]. Cohen et al. present a motion planner with motion primitives using heuristics, atomic actions, ARA* search, for a 7-DOF robotic manipulator [12]. Zhou et al. propose a robust and efficient quadrotor motion planning system for fast flight in three-dimensional complex environments with motion primitives [13]. Since the conditions required for each robot are different, each motion primitive was written considering the characteristics of each robot.

The purpose of this study is to find a suitable motion primitive for the VTT locomotion step for the given VTT configuration. In this paper, we set the costs and constraints necessary for VTT locomotion and then proceed with the task of deriving a motion primitive that suits it through an optimization process.

II. VARIABLE TOPOLOGY TRUSS

VTT is a modular truss structure composed of members and nodes. Each module is a truss member, and multiple modules can be connected together at a truss node. Each member contains a linear actuator that changes length using a spiral zipper. The nodes are hollow iron spheres that act as support points and connect the members. A spiral zipper is a mechanical element that can extend and retract by interlocking like a zipper [14], [15]. explain octahedron: rigid, symmetric, rolling For rolling locomotion, an octahedron shape is chosen. This shape has a total of 12 members and 6 nodes. The octahedron is symmetric and rigid structure, so it is strong against external forces. It also is easy to move by rolling on the ground because the distance nodes need to move is short. In this paper, the weight of each member is considered to be $1\,\mathrm{kg}$ and the nodes are considered to be $2\,\mathrm{kg}$ each.

III. ALGORITHM FOR MOTION PRIMITIVE OF LOCOMOTION STEP

A motion primitive is defined as a sequence of robot configurations for each time step, with each configuration consisting of all node positions. The algorithm of this study calculates the cost function over all possible motion steps for each time step and selects the one with the lowest cost to form the optimal configuration. The sequence of optimal configurations forms the motion primitive. The time step size

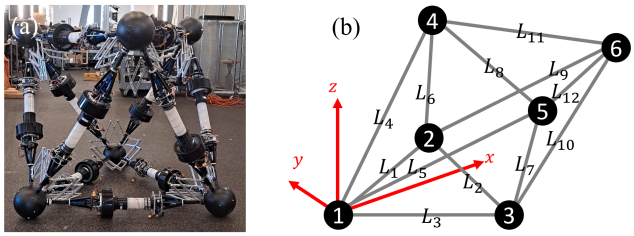

Fig. 1. (a) VTT prototype (b) Node and member labels

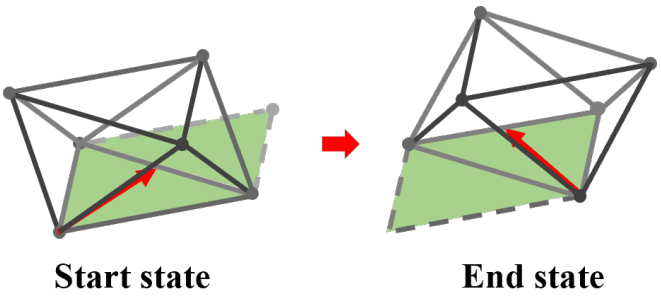

Fig. 3. Start state and end state of VTT

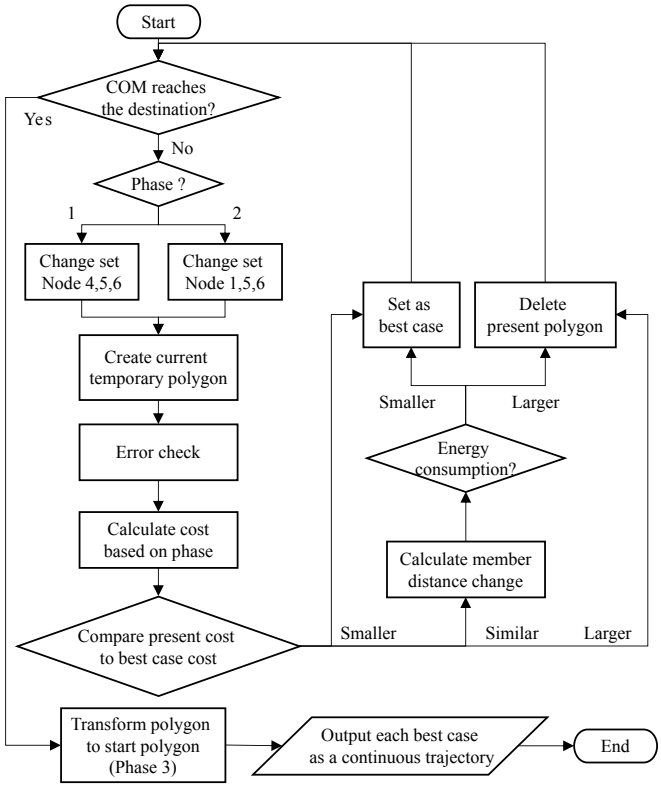

Fig. 2. Algorithm flowchart for calculating motion primitive of VTT

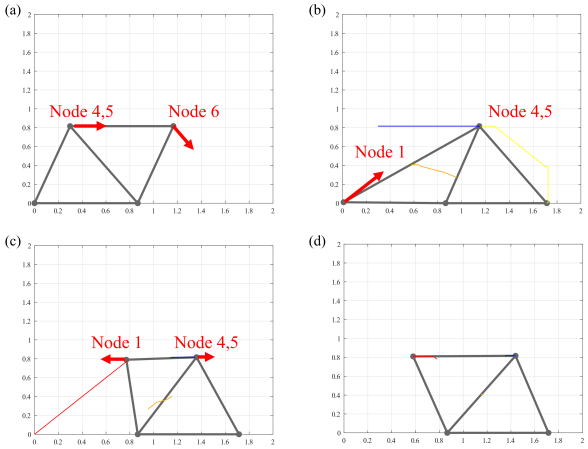

Fig. 4. Each phase of VTT locomotion. (a) Phase 1: Move node 6 to ground and node 4, 5 move forward. (b) Phase 2: Move CoM to next support polygon. Nodes 1, 4 and 5 move forward. (c) Phase 3: Reshape VTT. Nodes 1, 4 and 5 move in the shown direction. (d) End of locomotion.

is 50 ms. The whole process is described in the Figure 2. This section describes the algorithm.

In the start configuration, the length of all 12 members is 1 m each and node 1 is at the origin of the coordinate system. The step locomotion is in progress until the center of mass (CoM) of the VTT reaches the center of gravity of the triangle on the ground that is created via the locomotion step. So, when starting, it is checked whether the position of the CoM has reached the target point. If it has not yet been reached, the creation of possible displacements and cost evaluation is repeated in each configuration. After that, if the target CoM is reached, the shape of VTT returns to the initial shape with the length of each member at 1 m. As a result, the final arrival shape is the same as the starting shape, but only the position and direction change. It is illustrated in Figure 3.

A. *Phases of Locomotion*

One rolling step of locomotion is divided into three phases. In Phase 1, the algorithm puts node 6—the front node—on the ground, creating the next support polygon. The algorithm moves only node 4, 5, and 6 in Phase 1. In Phase 2, the CoM of VTT continues to move to the target point Nodes 4, 5, and 1, are moved in Phase 2. Phase 2 ends when CoM reaches the target position. Finally, in Phase 3, the VTT returns to the nominal shape. Once the robot returns to the nominal shape, a single locomotion step is completed, and this process can be continually repeated. The phases are illustrated in Figure 4.

B. *Generation of Possible Configurations*

On each time step, the algorithm generates many possible next configurations to consider. These are generated by considering all possible ways to move each node a small amount in all three axes. There are three cases of increasing, maintaining, and decreasing the x, y, and z coordinates of 6 nodes in each configuration by 1 cm. That is, the variables that change are $[p_1, p_2, p_3, p_4, p_5, p_6] = [(x_1, y_1, z_1), (x_2, y_2, z_2), ...(x_6, y_6, n_6)]$, where, for example, x_n means the x-coordinate value of node n. If there are no

constraints, $3^{6\times3}$ cases would be generated for each time step, yielding 387,420,489 possible configurations. Since this is a very large amount of computation, the amount of computation can be reduced by setting constraints.

- Fixed nodes: Nodes 2 and 3 are always fixed. So x_2, y_2, z_2 and x_3, y_3, z_3 do not need to be considered. Depending on the phase, one of nodes 1 and 6 is fixed. This means either x_1, y_1, z_1 or x_6, y_6, n_6 need to be considered, but not both.
- Symmetry: Nodes 4 and 5 are symmetric with respect to the xz-plane. This means (x_5, y_5, z_5) is the same as $(x_4, -y_4, z_4)$. Similarly, nodes 1 and 6 go through the center of the VTT. It means y_1 and y_6 are constrained to be 0.

As a result, the variables that need to be considered are x_1, z_1, x_4, y_4, z_4 or x_4, y_4, z_4, x_6, n_6. So the total number of cases on each time step is reduced to 3^5, that is, 243.

Out of the generated configurations, the ones that cause collision should be removed. Therefore, the next step is checking whether there are any collisions in the possible displacements created in this way. The only possible collision that can occur under these constraints is if the member between nodes 4 and 5 reaches zero length. Therefore, the algorithm checks whether nodes 4 and 5 intersect, and the corresponding displacements are removed from the selection.

C. Cost Function

After checking for collision, the cost is calculated according to the phase. The overall cost C is separated into locomotion cost C_l and shape cost C_s.

$$C = C_l + C_s \tag{1}$$

C_l is a major cost that affects the driving of VTT locomotion step. It is a weighted sum of two costs, C_d and C_{n_6}.

$$C_l = \begin{bmatrix} w_1 & w_2 \end{bmatrix} \begin{bmatrix} C_d \\ C_{n_6} \end{bmatrix} \tag{2}$$

C_d is the distance between the CoM and the target point, which is the center of the next support polygon. This cost factor is primarily used to drive the VTT towards the goal. The smaller this cost, the closer the center of gravity to the target point. C_{n_6} is distance between node 6 and the desired landing position of node 6. This cost drives node 6 towards the landing point in phase 1.

The shape cost C_s is not directly related to locomotion, but it is a cost to constrain the shape of the VTT. This limits large distortions of the truss.

$$C_s = \begin{bmatrix} w_3 & w_4 & w_5 & w_6 \end{bmatrix} \begin{bmatrix} C_{z_{1,4}} \\ C_{y_{4,5}} \\ C_{x_{1,4}} \\ C_{Dis} \end{bmatrix} \tag{3}$$

$C_{z_{1,4}}$ is height between node 1 and 4. This cost penalizes node 4 from moving too high during phase 1 and encourages node 1 to lift off the ground during phase 2. Since nodes 4 and 5 are constrained to be symmetric, no additional cost is necessary for node 5. $C_{y_{4,5}}$ is the difference between the distance between nodes 4 and 5 and the nominal length. It maintains the correct distance between nodes 4 and 5. $C_{x_{1,4}}$ is the difference between the distance between the x-positions of nodes 1 and 4 and the desired offset. This cost is only applied in phase 2. C_{Dis} is distance between CoM and nodes 1, 4 and 5. It serves to hold nodes 1, 4, and 5 so that they do not deviate too much from the center.

The total cost is calculated by multiplying each factor and weight numbers. The weights differ between phase 1 and phase 2. In phase 1, $[w_1, w_2, w_3, w_4, w_5, w_6] = [1.2, 1, 0.2, 0.2, 0, 0.1]$ are used as weights, while in phase 2, $[w_1, w_2, w_3, w_4, w_5, w_6] = [1.2, 0, 0.2, 0.2, 0, 0.1]$ are used as weights. Since important factors are different for each phase, unnecessary cost terms are excluded depending on the phase. In phase 3, the cost calculation is all omitted because there is only the length of the member changes for reshaping VTT to the starting shape.

After calculating the cost, the algorithm finds the case with the lowest cost. If multiple cases have similar costs, the best case is selected by comparing the energy consumption rate in addition to the cost. Energy consumption increases with the rotation of the motor inside the member. Therefore the sum of the changed length of each member is compared and the case with the least change is selected.

At each time step, the best case configuration is selected and continuously collected. The algorithm checks if the CoM has arrived near the target CoM point. If it has not arrived yet, the locomotion has not been completed and the process is repeated. Once it arrives, the sequence of configurations is output as the motion primitive and the algorithm ends.

IV. RESULT

Through this algorithm, an optimized motion primitive was derived. As a result of the simulation, the resulting motion primitive was composed of a total of 195 configurations and is shown in Figure 5. Changes in the position of each node are indicated in Figure 6 except for nodes 2 and 3, which are fixed nodes. Each phase is separated by a vertical gray dotted line on the figure.

In the change of member length for each member, duplicates are excluded because members 1, 4, 6, 9, and 11 are the same as members 3, 5, 7, 10, and 12 respectively due to the symmetry condition. In this section, we will analyze whether this motion primitive is appropriate.

The CoM for each configuration of the derived motion primitive is depicted in Figure 7. The CoM moves from (0.577, 0, 0.408) to (1.155, 0, 0.408) point, from the starting point to the target point. As expected, the y value remains 0, and the x value continuously increases. Since the z value moves between the highest point of 0.407 and the lowest point of 0.269, it can be said that the CoM moved to the target point without much change. So, this motion primitive can be seen to move without much unnecessary movement in terms of the motion of the CoM.

To check whether the algorithm works, it is necessary to trace the cost. The cost for each configuration is depicted

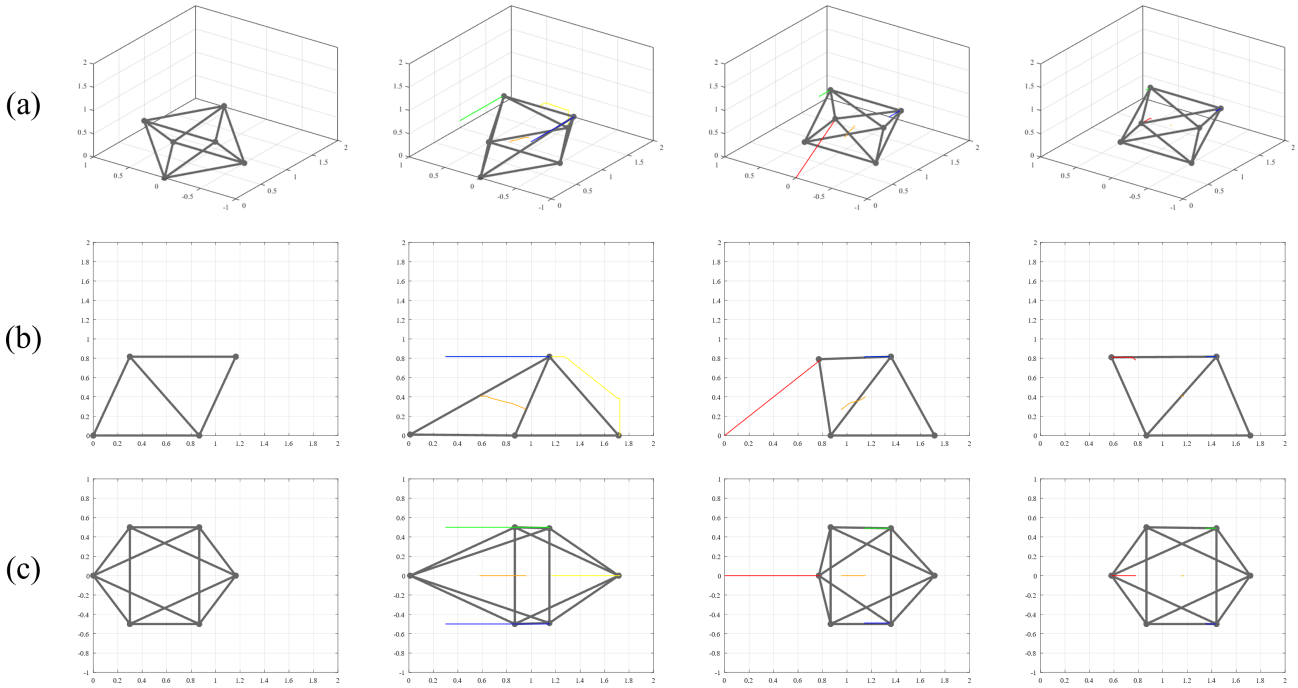

Fig. 5. Motion primitive derived by algorithm - VTT locomotion is shown in (a) 3D view, (b) side view, and (c) top view

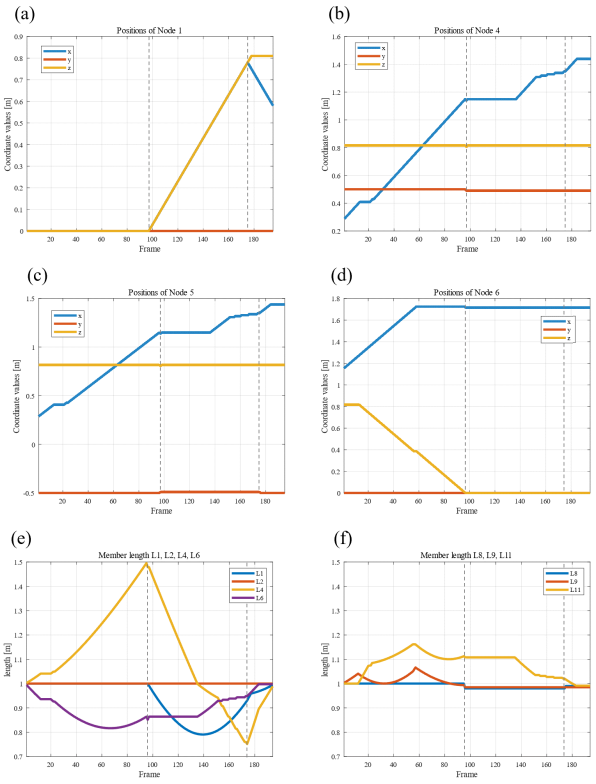

Fig. 6. Trajectory of (a) node 1, (b) node 4, (c) node 5, (d) node 6, and length of (e) members 1, 2, 4, 6 and (f) members 8, 9, 11

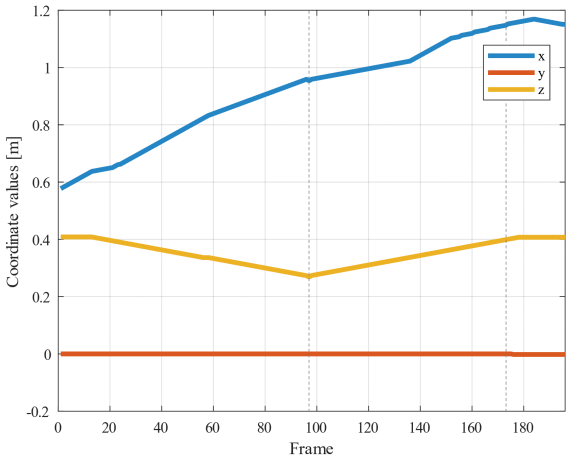

Fig. 7. Trajectory of center of mass

in Figure 8. The cost decreases overall, but changes significantly when the phase changes. The cause seems to be the change in cost weights depending on the phase. Therefore, it is necessary to analyze whether this large change in cost affects the actual locomotion by checking other factors.

In the algorithm, the distance between the CoM and the target CoM point is set as a major factor of locomotion. Therefore, the distance between the CoMs during locomotion, which is shown in the Figure 9, was checked. The CoM continuously decreases without unnecessary increase overall, and it was found that the value bounced when the cost weighting changed due to a change of phase, but the

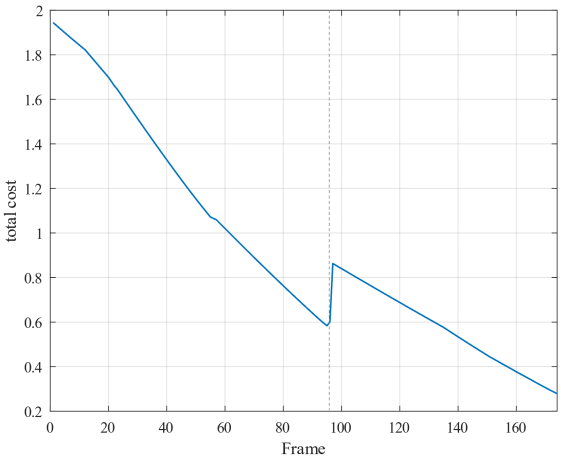

Fig. 8. Cost value on motion primitive

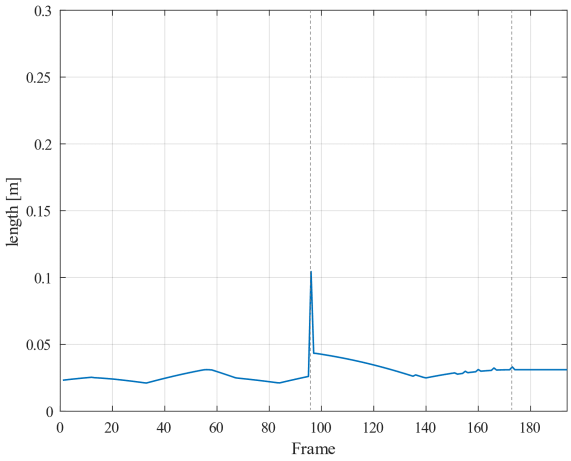

Fig. 10. Sum of member length changes

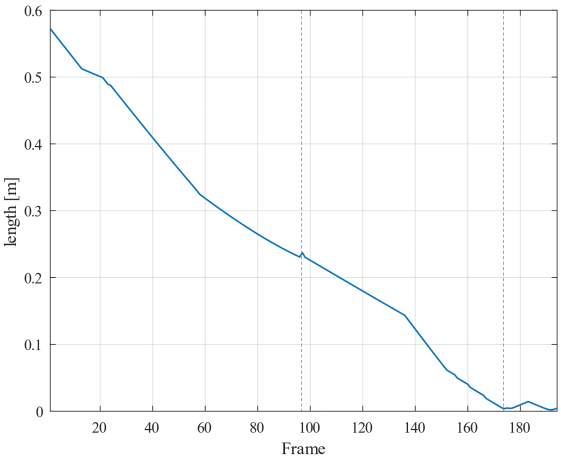

Fig. 9. Distance between present and objective CoM

difference in value was $0.007\,\mathrm{m}$, which is small enough to be ignored. By checking the motion of the CoM, it was judged that there was no big problem in the algorithm.

Energy consumed during this motion primitive can be approximated according to the member length changed for each configuration. The locomotion of the VTT consists of the length change of the member. The energy used for changing of the member lengths can measure the energy consumption of the VTT. In other words, minimizing the length change improves the energy efficiency of the motion primitive. The sum of the length changes of all the members for each configuration is shown in Figure 10. The average of the changed lengths of the 12 members in the sequence of configurations is $0.029\,\mathrm{m}$. The sum of the total changed lengths was $5.61\,\mathrm{m}$. The energy consumption of one member moving $0.47\,\mathrm{m}$ during step locomotion is reasonable.

In summary, the motion primitive derived by the algorithm can work in reality without unnecessary motion. There is no discontinuous or abrupt change in the position of the nodes and the lengths of the members, and the CoM also approaches the target point continuously. Although a slight jump during phase change was observed, it does not significantly effect the actual member length change and node position change. Therefore, it was judged that this motion primitive is sufficiently effective and can be applied in reality.

V. CONCLUSION

In this paper, an algorithm for deriving the motion primitive of step locomotion for VTT is presented. This algorithm achieves locomotion by moving the CoM of the entire VTT to the target point. As a result, the derived motion primitive can be applied to an actual VTT given the node positions, member lengths, cost, and distance between CoM.

In future work, the algorithm can be improved in two ways. The first is to add more physical characteristics of VTT so that many factors are considered by the algorithm. For example, collision between nodes and the ground, prevention of damage of spiral zippers due to friction and impact, and deflection of the VTT according to self-weight. Considering these, it will be possible to proceed with locomotion with less burden on the VTT. The second way is to improve the computational efficiency. There is already a significant amount of computation with the current algorithm, and the amount of computation was reduced through various conditions. However, if several factors are added to this algorithm, the amount of computation inevitably increases. Therefore, future works are needed to reduce the amount of computation by further researching the search method.

ACKNOWLEDGMENT

This research was supported by the MOTIE (Ministry of Trade, Industry, and Energy) in Korea, under the Human Resource Development Program for Industrial Innovation(Global) (P0017306, Global Human Resource Development for Innovative Design in Robot and Engineering) supervised by the Korea Institute for Advancement of Technology (KIAT)

REFERENCES

[1] K. Miura, "Design and operation of a deployable truss structure," in *NASA. Goddard Space Flight Center The 18th Aerospace Mech. Symp.*, 1984.

[2] K. Miura, H. Furuya, and K. Suzuki, "Variable geometry truss and its application to deployable truss and space crane arm," *Acta Astronautica*, vol. 12, no. 7-8, pp. 599–607, 1985.

[3] P. C. Hughes, W. G. Sincarsin, and K. A. Carroll, "Trussarm—a variable-geometry-truss manipulator," *Journal of Intelligent Material Systems and Structures*, vol. 2, no. 2, pp. 148–160, 1991.

[4] J. Sastra, S. Chitta, and M. Yim, "Dynamic rolling for a modular loop robot," *The International Journal of Robotics Research*, vol. 28, no. 6, pp. 758–773, 2009.

[5] W. H. Lee and A. C. Sanderson, "Dynamic rolling locomotion and control of modular robots," *IEEE Transactions on robotics and automation*, vol. 18, no. 1, pp. 32–41, 2002.

[6] E. Park, J. Bae, S. Park, J. Kim, M. Yim, and T. Seo, "Reconfiguration solution of a variable topology truss: Design and experiment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1939–1945, 2020.

[7] A. Spinos, D. Carroll, T. Kientz, and M. Yim, "Topological reconfiguration planning for a variable topology truss," *Journal of Mechanisms and Robotics*, pp. 1–33, 2021.

[8] C. Liu, S. Yu, and M. Yim, "Motion planning for variable topology truss modular robot," in *Proceedings of Robotics: Science and Systems*, 2020.

[9] S. Park, J. Bae, S. Lee, M. Yim, J. Kim, and T. Seo, "Polygon-based random tree search planning for variable geometry truss robot," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 813–819, 2020.

[10] J. Bae, S. Park, M. Yim, and T. Seo, "Polygon-based random tree search algorithm for a size-changing robot," *IEEE Robotics and Automation Letters*, 2021.

[11] K. Hauser, T. Bretl, K. Harada, and J.-C. Latombe, "Using motion primitives in probabilistic sample-based planning for humanoid robots," in *Algorithmic foundation of robotics VII*. Springer, 2008, pp. 507–522.

[12] B. J. Cohen, S. Chitta, and M. Likhachev, "Search-based planning for manipulation with motion primitives," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 2902–2908.

[13] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3529–3536, 2019.

[14] C. Liu, A. Bera, T. Tsabedze, D. Edgar, and M. Yim, "Spiral zipper manipulator for aerial grasping and manipulation," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 3179–3184.

[15] F. Collins and M. Yim, "Design of a spherical robot arm with the spiral zipper prismatic joint," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 2137–2143.