Variable Geometry Truss Robot Motion Planning on Irregular Terrain

OZGUR GULSUNA
Middle East Technical University, Ankara, TR

Abstract

This paper explores Variable Geometry Truss (VGT) systems, emphasizing their kinematics and path planning for irregular terrains. Utilizing an octahedral topology, the study establishes kinematic relationships and addresses rigidity considerations. Introducing the Polygonal Random Tree for Irregular Terrains (PRT-IT) algorithm, the paper demonstrates its effectiveness in navigating complex landscapes, avoiding obstacles, and adapting to slope constraints. A size control mechanism mitigates foot placement errors, enhancing the robot's adaptability. Simulation results validate the PRT-IT algorithm's success in diverse terrains, making a contribution to VGT systems' understanding and advancement in robotic motion planning and control.

***Index Terms*— Motion Planning, Irregular Terrain, PRT, PRT-IT, Probabilistic Methods, Shackleton Crater, Terrestrial Bodies.**

I. INTRODUCTION

The Variable Geometry Truss (VGT) systems, composed of linearly actuated truss members and passive rotational joints, offer adaptability across wide range of applications. These robotic systems have the capability to alter their shape and topology significantly through coordinated actuation of their linear members. The foundational concept of VGT systems was introduced by Miura et al., with a primary focus on fixed configurations with crane structure [1]. Subsequent advancements in the field involved the exploration of VGT as a modular and mobile system by Hamlin et al. [2], expanding the scope of potential applications. Furthermore, Usevitch et al. contributed to the literature by introducing kinematic analysis and optimization-based locomotion for VGT systems [3], thereby enhancing our understanding of the dynamic capabilities of these robotic structures. The control of robots with a large degrees of freedom is important leverage that allows robots to adapt to the wide variety of tasks.

The prior research focused on various aspects of the system, with an emphasis on locomotion. The problem addresses the kinematic relationships to determine a series of size changes in member lengths that enable the desired motion of the whole structure. In the case of complex truss structures, the rigidity of the topology and mechanical constraints contribute to a large solution space. Consequently, solving this problem necessitates the optimization-based methods. The present work builds upon the foundation, incorporating kinematic relations and derivations. Specifically, the kinematic relations and optimization methodology stems from the insights provided by Usevitch et al. [3].

The second facet of research concerning VGT systems involved path planning from a higher-level perspective, with the objective of navigating the environment to reach the goal position. Existing literature in this domain concentrated on employing probabilistic methods, with a specific emphasis on the Polygonal Random Tree search (PRT) algorithm, forming the foundational basis for the current study. It is noteworthy to highlight that prior work on path planning focused strictly on scenarios involving planar terrains and well-defined obstacles.

The objective of this study is to extend the current understanding of VGT systems by advancing both the kinematic relations and the nonlinear optimization problem, in terms of adapting it for irregular terrain scenarios. Additionally, from a higher-level planning perspective the PRT algorithm is aimed to be expanded to address challenges posed by more complex terrains and the obstacles laying coherently in real-world environments.

II. VARIABLE GEOMETRY TRUSS SYSTEM

In the context of a network of linear actuators within a Variable Geometry Truss (VGT) system, determining node positions by manipulating edge lengths is a non-trivial task. Each link length imposes a constraint on the node positions, and the challenge is to find positions that satisfy all these constraints, accounting for translation and rotation of the entire network. The rigidity of the underlying graph plays a pivotal role, with different classes of solutions emerging based on this rigidity. For instance, if the system of equations yields infinite solutions, indicating non-rigidity, the framework allows movement relative to itself without violating length constraints. Conversely, a rigid framework possesses a discrete number of solutions, and any deflection of the system relative to itself violates the length constraints. A simple example on the rigidity is a parallelogram with passive joints, this mechanism is under-constrained and the solution for the edge lengths result in variety of node positions. However if one connect two non-neighbouring edges then with the new length constraint the system becomes rigid. The analysis of device kinematics and system constraints in subsequent sections will be an important step in terms of tacking the robot locomotion task.

A. Overview

The chosen topology for the VGT robot is an octahedron, consisting of 6 nodes and 12 members and support polygon highlighted with red, as illustrated in Figure 1. The robot's structure can be represented as a graph $\mathbf{G}$, which comprises a node set $\mathbf{N} = \{n_1, n_2, \ldots, n_6\}$ referring to the vertices of the robot. The connectivity or adjacency set, denoted as $\mathbf{A}$, wires specific nodes together, defining the structure through elements like $\{\{n_1, n_2\}, \{n_2, n_3\}, \ldots, \{n_6, n_4\}\}$. Additionally, the member length set, denoted as $\mathbf{L}$, serves as an analogous representation for the edges of the robot, with $\mathbf{L} = \{l_1, l_2, \ldots, l_{12}\}$ which are the edge lengths. The nodes in three-dimensional matrix form are expressed as $\mathbf{n_i} = \begin{bmatrix} n_{i_x} & n_{i_y} & n_{i_z} \end{bmatrix}$, where each n_i represents a node. Lastly, the vector $\mathbf{x}$ denotes the position of all the nodes $\mathbf{x} = \begin{bmatrix} n_{1_x} & \ldots & n_{6_x} & n_{1_y} & \ldots & n_{6_y} & n_{1_z} & \ldots & n_{6_z} \end{bmatrix}$ which will come in handy in Jacobian construction.

B. Kinematics

The kinematic structure of the robot is characterized with the relationships emanating from the distances between two connected nodes defined as:

$$L_k = \|n_i - n_j\| \; \forall \{n_i, n_j\} \in N$$

Expanding this relation into matrix form results in the representation:

$$L = Rx$$

This matrix equation is recognized as the inverse kinematics equation in matrix form. The term "inverse kinematics"

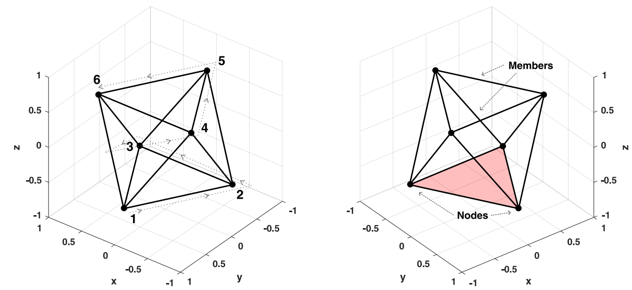


Fig. 1: Octahedral topology of the VGT robot and the node order.

can be used since the configuration of the node points is analogous to the end effector position for a serial manipulator, wherein three-dimensional coordinates are utilized to locate the nodes, and the lengths of the members correspond to the joint positions again in a serial manipulator. The matrix representation of the topology is expressed as:

$$\begin{bmatrix} L_1 \\ L_2 \\ \vdots \\ L_{12} \end{bmatrix}_{12\times1} = \begin{bmatrix} R_{1-1}(x) & R_{1-2}(x) & \ldots & R_{1-18}(x) \\ R_{2-1}(x) & R_{2-2}(x) & \ldots & R_{2-18}(x) \\ \vdots & \vdots & \ddots & \vdots \\ R_{12-1}(x) & R_{12-2}(x) & \ldots & R_{12-18}(x) \end{bmatrix}_{12\times18} \begin{bmatrix} x_1 \\ \vdots \\ x_6 \\ y_1 \\ \vdots \\ z_6 \end{bmatrix}_{18\times1}$$

Here matrix $R(x)$ is a Jacobian matrix relating the specific nodes to a norm function, constructing a matrix. There is however a specific consideration on this Jacobian, is which the function depends on the current configuration of the robot, in more detailed explanation the R_{1-1} depends on $x_1, x_2, y_1, y_2, z_1, z_2$ for which the the matrix changes for each calculation.

The octahedral configuration is considered for its rigidity. Once the six degrees of freedom in space are constrained, the robot maintains a fixed position. The structural rigidity of the octahedron can be demonstrated using the linear system equations.The set of equations is provided above, with 12 equations and 18 unknowns for the system, resulting in 6 degrees of freedom. This corresponds to the 6 degrees of freedom in three-dimensional solids; hence, once those are constrained, the rest of the system becomes rigid.

Differential kinematics are particularly important as they link changes in node positions to actuation in member lengths within the set of connections. The equation governing this relationship is:

$$\dot{L} = R\dot{x}$$

This matrix equation shows how changes in node positions ($\dot{x}$) are connected to adjustments in the member lengths configuration ($\dot{L}$), where $R(x)$ is the matrix known as the Jacobian for this system.

The last key aspect of the system involves the center of mass and its projection onto the support polygon. The center of mass represents the robot's position in three-dimensional space using just one point. Its projection onto the ground ensured that it stays within the support polygon, defines the robot's stability. Consequently, the center of mass trajectory have a significance on higher-level planning tasks and it is needed to considered.

The center of mass of the robot is calculated as if all the weight is collected around the nodes, hence the center of mass position ($\overline{x}$) is the mean of its node positions. The projection of the center of mass onto the support polygon and averaging the node positions can be depicted with a single projection matrix that is called $\mathbf{M}$ and this matrix changes as the robot steps over to another support polygon.

$$x_{cm} = Mx$$

The result of the path planning it a trajectory of the projections of the center of mass, hence it is known. Then one needs to relate back to the set of change of member lengths from the center of mass position in order to result in which actuator is expanded or contracted. We can formulate the whole problem relating center of mass projection to the member lenghts with:

$$\dot{x}_{cm} = MR^{-1}\dot{L}$$

The problem at hand can be formulated in the form $b = Ax$, where $\dot{L}$ is the unknown. However, a critical challenge arises as the Jacobian matrix needs to be inverted. This presents a crucial condition as the Jacobian is not a square matrix, necessitating the use of a pseudo-inverse. The resulting inverse is a tall matrix. When multiplied by the projection matrix, which condenses the 18-dimensional vector into 3 dimensions, the resulting matrix is of size 3 by 12, characterized as a wide matrix. This wideness implies that the system possesses infinitely many solutions, indicating that numerous configurations of member lengths exist for a single center of mass projection.

Optimization becomes particularly significant in this scenario. By narrowing down the solution space through constraints such as member lengths, fixed nodes, and collision checks, it is possible to identify a solution that minimizes a certain aspect of the system. In this context, minimizing the norm of the nominal length of the members proves advantageous. This minimization serves two purposes: it maintains a constant size for the VGT robot throughout its movement, and it minimizes the excitation of each actuator, promoting more uniform motion.

$$min\ J(x) := \|L(x) - L_{nom}\|^2$$

In the context of this problem, linear programming (LP) is applicable as an optimization technique to find the values of variables (in this case, member lengths) that minimize an objective function while satisfying a set of linear equality and inequality constraints. The LP formulation allows for identification of an optimal or near-optimal configuration for the robot's member lengths. This can contribute to achieving a stable motion.

III. IRREGULAR TERRAIN PATH PLANNING

A. Terrain Processing

The implementation process involves the selection of a terrain information base which is the structure that the information comes in, this data could be in the form of images, a depth map, a point cloud, generated mesh etc. The most widely used type, which is derived from laser range distance sensors, is a point cloud data. For the remote operation of unexplored places, an onboard sensor mostly be a kind of a lidar type. The algorithm should effectively handle this type of input. The workflow for terrain information processing can be divided into three main steps: Input processing, Point cloud generation, and Mesh formations.

The chosen input method for terrain information is a depth map. This type of data is already available since it is easily obtained for terrestrial objects such as planets, asteroids, and the moon. This depth image can also be altered by drawing on it to introduce new obstacles, large pits, or structures for the experimentation phase.

In the second step, point cloud information is extracted from the input image, in this very step down-sampling the image is easily done to work with less number of points. If point cloud data is already available, then this process is bypassed, and only the point information is utilized.

The final step involves mesh generation from the point cloud, with a defined detail level. Higher detail results in a smaller mesh size but increased count. Within the measurement error range of the lidar sensors, the mesh information is generally adequate in size, avoiding excessive detail.

The workflow and outcomes of this process with samples are illustrated in the Figure 2.

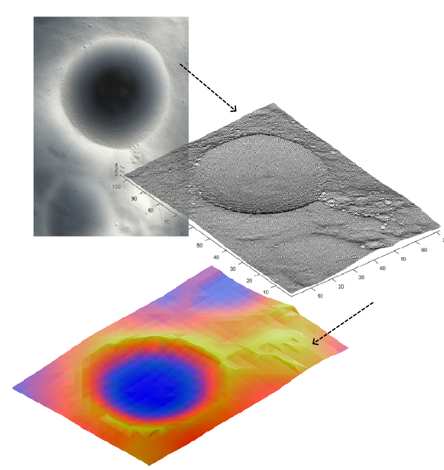


Fig. 2: Terrain processing steps visualized.

B. PRT-IT: Polygonal Random Tree Search for Irregular Terrain

The Polygonal Random Tree (PRT) search algorithm has been studied in the literature, but its design is limited for planar surfaces, hindering its applicability. This paper aims to extend the methodology to accommodate irregular terrains and potentially address path planning over closed mesh structures and multi-level environments.

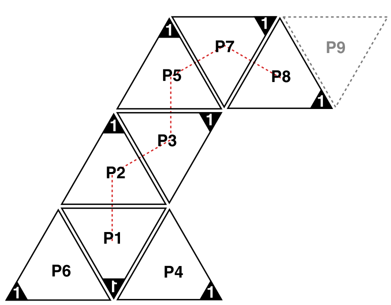
The proposed extension, termed PRT-IT, utilizes a tree structure comprised of triangle polygons, aligning their size with the support polygons for the VGT robot. Connections exist between the new polygon and its parent polygon within the tree list structure. The convention on naming of the polygons and respective node ordering methodology is depicted in Figure .

In each iteration of the algorithm, an objective point is selected to guide the robot's movement. Typically, this objective point is randomly sampled within the environment, enabling the robot to explore different locations on the map. Alternatively, at times, the objective point is selected as the goal point. The ratio between randomly sampled points and goal points is referred to as the mixing factor. This factor plays a pivotal role in balancing the exploration strategy.

A higher mixing factor tends to make the robot act greedily, heading directly towards the goal without exploration. Conversely, a lower mixing factor encourages more thorough exploration of the map but may extend the time required to find a solution. Striking a suitable balance with the mixing factor is desired however a wide range of mixing factors work adequately. For this study the mixing factor is taken as 10%.

Polygonal Tree Structure

$P_1 \leftarrow$
$P_2 \leftarrow P_1$
$P_3 \leftarrow P_2$
$P_4 \leftarrow P_1$
$P_5 \leftarrow P_3$
$P_6 \leftarrow P_1$
$P_7 \leftarrow P_5$
$P_8 \leftarrow P_7$
$\vdots \leftarrow \vdots$



The initial node of a polygon is denoted as "1" and is identified as the farthest node from the preceding polygon; this is also commonly referred to as the "foot." Subsequent nodes are ordered following the right-hand convention. The edges within the polygon are named in relation to the opposing nodes, consistently adhering to the right-hand rule. The tree structure for the algorithm is exemplified below. It's noteworthy that the edges neighboring the parent polygon are designated as the first edges, maintaining the right-hand convention. This representation illustrates how the algorithm organizes nodes and edges within a polygon.

Algorithm 2 PRT-IT Algorithm

function PRTPLANNER($p_{\text{init}}, s_{\text{goal}}$, size)
    $goal\_reached$ = false;
    $mixing\_factor = 0.1$;
    $tree \leftarrow$ struct($'polygon', [], 'parent', []$);
    while $\neg goal\_reached$ **do**
        if rand $\geq mixing\_factor$ **then**
            $s_{\text{obj}} \leftarrow s_{\text{rand}}$;
        else
            $s_{\text{obj}} \leftarrow s_{\text{goal}}$;
        end if
        $temp\_polygon \leftarrow$ findNearest(s_{obj}, size);
        if temp_normal_angle < slope_limit **then**
            $tree$(end + 1) $\leftarrow temp\_polygon$;
        end if
        if closest_distance < sigma **then**
            $goal\_reached = 1$;
        end if
    end while
    $solved\_tree \leftarrow []$;
    while $parent\_polygon \neq 0$ **do**
        $solved\_tree$(end) $\leftarrow tree(parent\_polygon)$;
    end while
    $solved\_tree \leftarrow$ flip($solved\_tree$);
end function

Once an objective point is determined, the function $findNearest$ iterates over all polygons in the tree, extends new polygons from the two edges not belonging to the parent polygon. While these extended polygons may not necessarily touch the surface due to the non-planar terrain, the process efficiently checks the distance between the objective point and temporary foot points, to find the temporary foot that is closest to the objective.

This polygon alternative with the minimum distance is then selected the candidate for this iteration. However, further evaluation of the polygon before appending it to the tree structure seeks proper contact with the ground for the temporary foot, given that the robot needs to place its foot on a surface. Consequently, a line is emitted from the temporary foot, and the intersection point of this line with the surface mesh is identified. This intersection point serves as the new foot position, and a support polygon is formed, consisting of the new foot in contact with the ground and two old feet belonging to the parent.

However, it's crucial to note that this new polygon is initially considered a candidate. A subsequent check examines its normal angle to ensure that the angle at which the new polygon is situated is not excessively steep, it it is then the new polygon is discarded. This precautionary measure is taken to avoid situations where the robot might slide down or face potential endangerment due to the terrain's steepness. This measure also considers all steep features of the terrain as an obstacle and avoids those.

The algorithm iterates repeatedly, selecting a new objective point at each iteration until the goal point is in close proximity to the last polygon. Upon reaching this condition, the iteration is halted, and the backtracking process commences using the tree structure. Backtracking involves following the parent connections until the root polygon is reached. The final outcome is a single list of polygons, constituting the solution to the path planning problem. This solution ensures the avoidance of obstacles and adheres to specified limitations, guiding the robot to the goal.

C. Size Control

The foot position selection from the mesh involves emitting a line from an temporary foot, however this results in error in foot placement since the reach of the robot for the same dimensions is circular rather than a line. This error term is particularly crucial when dealing with nonplanar surfaces, where inaccuracies in foot placement could accumulate, leading to a gradual increase in the robot's size, especially evident on larger slopes.

To mitigate this issue, the intersection of the surface mesh and foot reach trajectory can be found, however this will result in two potential solutions. Subsequently, an additional check is needed to determine which intersection is closer.

Alternatively, an effective solution to the problem of unintended robot enlargement involves the implementation of a gain factor for controlling the robot's size. This approach entails providing a reference size as an input to the algorithm, calculating an error term for the new polygon. If the distance between two previous foot points exceeds the reference size, the gain term is negative, causing the robot to shrink. Conversely, if the distance is smaller, the gain factor becomes positive, leading to an increase in the robot's size. The gain factor serves as a proportional controller for the robot's dimensions, offering a dynamic mechanism for size adjustment. Furthermore, the reference size can be modified within the search, providing adaptability to different scenarios.

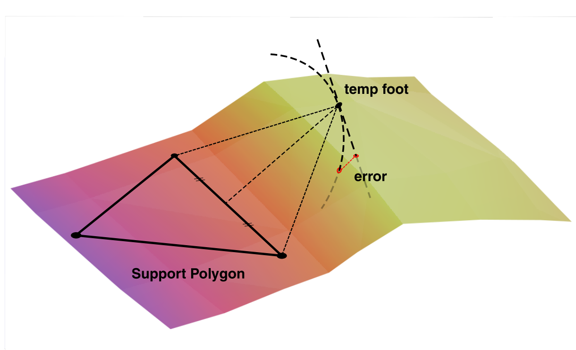


Fig. 3: The approximated foot position and the ground truth. The error term is always positive, hence the robot tend to scale up.

IV. Simulation Results and Discussions

Simulations have been conducted to validate the PRT-IT algorithm across diverse irregular terrains. The initial experiment puts the algorithm to the test in a challenging environment with multiple stressors. The map features a dense crater system characterized by substantial irregularities in the terrain. Notably, a deep crater is situated on the right side of the map Fig. 4. In this scenario, the VGT robot decides not to descend into the deep crater. Instead, it rotates around it, showcasing the algorithm's capacity to navigate complex landscapes and make strategic decisions to avoid potential obstacles.

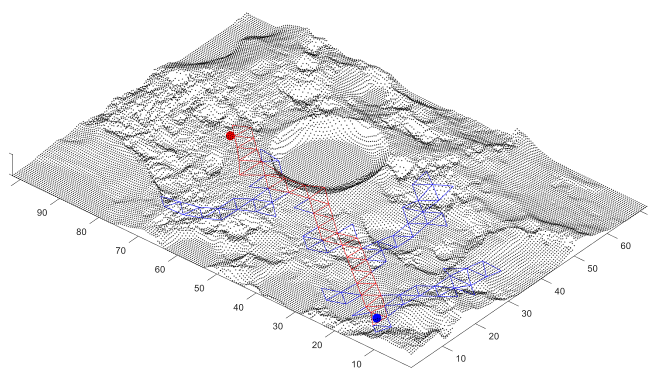


Fig. 4: The terrestrial test on a dense crater, red is goal, blue is initial point.

In the subsequent environment, the focus shifts to another crater system, specifically the Shackleton Crater on the moon. The experiment involves varying the obstacle slope limit, which determines the allowable angle for the terrain structure. In the initial test, a relatively high slope limit is set, causing the algorithm to disregard the irregular terrain as an obstacle. Conversely, in the second test, the slope limit is reduced. In response, the algorithm strategically chooses a path that avoids steeper slopes, showcasing its adaptability to different environmental conditions and its ability to adjust its trajectory based on specified slope constraints.

The significance of the effective slope constraint lies in its capability to discern various geological features as obstacles or non-obstacles. This functionality can serve as a foundational component in the development of an intelligent system. Such a system could characterize the soil and make informed decisions regarding the appropriate slope constraint based on its understanding of the terrain.

Another noteworthy aspect of slope-based obstacle avoidance is that obstacles are more softly bounded. As the VGT system is enlarged, geological features become relatively smaller in comparison. Consequently, the slope criteria become more easily satisfied. This observation highlights the adaptability of the algorithm to changes in the scale of the VGT system. It suggests

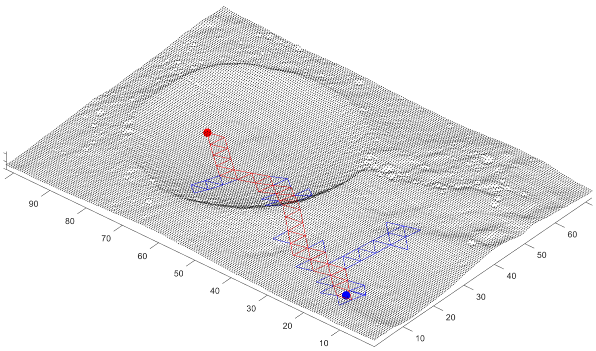


Fig. 5: Slope constraint is eased hence the path is straight.

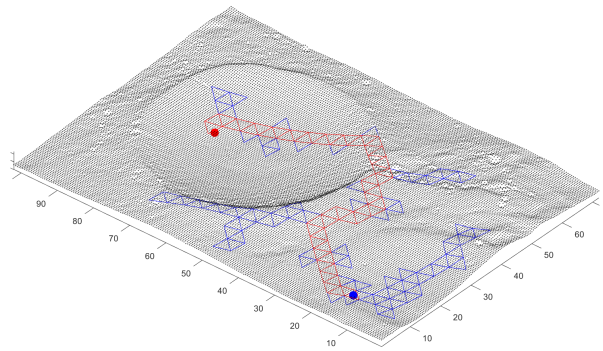


Fig. 6: Slope constraint is reduced, hence the rerouting is necessitated.

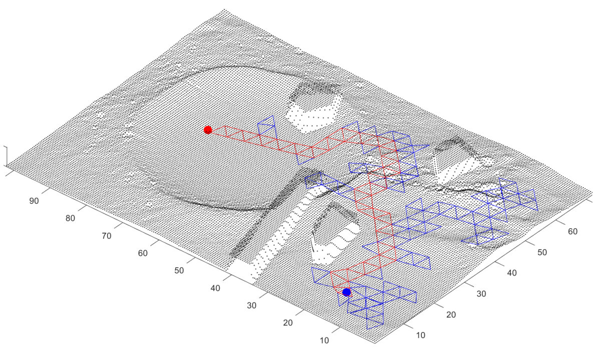


Fig. 7: Sharp features as obstacles added to the map, the VGT traverses around those.

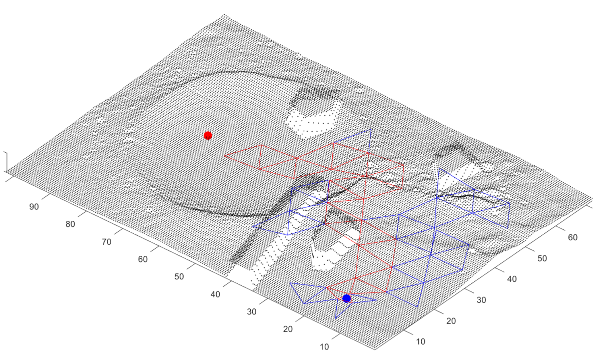


Fig. 8: As the robot is enlarged it can navigate over them, stepping over in a way.

that, with an increase in the system's size, the algorithm can effectively navigate through terrains with the same geological features, demonstrating a degree of scalability and versatility in addressing different scales of robotic systems and associated terrains.

V. Conclusions

In conclusion, this paper presents a comprehensive exploration of Variable Geometry Truss (VGT) systems, focusing on their kinematics and path planning for irregular terrains.

The study extends the application of VGT systems to irregular terrains. The Polygonal Random Tree (PRT) search algorithm is adapted and extended to create the PRT-IT algorithm, designed for irregular terrains. The algorithm demonstrates its capability to navigate complex landscapes, avoiding obstacles, and adapting to varying slope constraints.

Moreover, the paper introduces a size control mechanism for the robot, addressing issues related to foot placement errors on non-planar surfaces also serving as a controller for dynamically adjusting the robot's dimensions during the search process.

Simulation results validate the effectiveness of the PRT-IT algorithm across diverse irregular terrains. The algorithm successfully navigates through challenging environments, avoiding obstacles and adjusting its trajectory based on slope constraints.

In summary, this paper contributes to the understanding and advancement of VGT systems, providing insights into their kinematics, control, and path planning for irregular terrains. The proposed PRT-IT algorithm offers a promising solution for intelligent navigation in complex environments, laying the groundwork for further developments in the field of robotic motion planning and control.

REFERENCES

[1] Variable geometry truss and its application to deployable truss and space crane arm
vol. 12, no. 7, pp. 599–607, 1985.

[2] G. Hamlin and A. Sanderson
Tetrobot: a modular approach to parallel robotics
IEEE Robotics Automation Magazine, 1997.

[3] N. S. Usevitch, Z. M. Hammond, and M. Schwager
Locomotion of linear actuator robots through kinematic planning and nonlinear optimization
IEEE Transactions on Robotics, vol. 36, no. 5, pp. 1404–1421, 2020.