Planning for a Tight Squeeze: Navigation of Morphing Soft Robots in Congested Environments

Edward Gough[1], Andrew T. Conn[2] and Jonathan Rossiter[3]

***Abstract*—Autonomous navigation methods can prevent robots becoming trapped between obstacles and ensure that they take the most efficient route. As mobile robots have a limited power supply, selecting the most efficient route is crucial. This paper presents a path-planning method for morphing soft robots in congested environments. The proposed method is suitable for all scales of robots and environments, from intra-organ biomedical navigation to search-and-rescue operations in cave networks. The method utilizes 3D Voronoi diagrams and Dijkstra's algorithm to calculate an optimal path that balances cost between the size and shape change of the robot and the length of the path. The Voronoi method is particularly suitable for circumferentially expanding robots because the waypoints generated lay where a device with a circular cross-section would naturally sit. The method is demonstrated by simulation in procedurally generated environments with either spherical or continuous obstacles to illustrate the effectiveness of the method for in-situ planning and as an aid to design. This paper provides a generic approach that has the potential to be easily adapted for many applications across healthcare, industry and space exploration.**

***Index Terms*—Modeling, Control, and Learning for Soft Robots, Motion and Path Planning**

I. Introduction

SOFT robots are able to deform around structures that would trap a traditional rigid robot, making accessing confined and congested environments one of their key applications [1]. The broad range of applications for soft robots [2] has led to a variety of forms and locomotion modes being developed [3]–[6]. Despite this aptitude, path-planning methods designed for morphing soft robots have received little attention. Many soft robots can be controlled directly and exploit adaptive gaits for their specific form of locomotion [7], [8]. However, bespoke path-planning for these robots remains absent.

Manuscript received: October, 15, 2020; Revised January, 21, 2021; Accepted February, 20, 2021.

This paper was recommended for publication by Editor Dr. Cecilia Laschi upon evaluation of the Associate Editor and Reviewers' comments. E.G. was supported by the EPSRC Centre for Doctoral Training in Future Autonomous and Robotic Systems (FARSCOPE) and BT Applied Research through the EPSRC iCASE scheme. J.R. was supported through EPSRC research grants EP/S026096/1, EP/R02961X/1 and EP/M020460/1, and by the Royal Academy of Engineering as a Chair in Emerging Technologies. A.T.C. was supported by EPSRC grant EP/R02961X/1

[1]E.G. is with the Bristol Robotics Laboratory, Bristol, UK edward.gough@bristol.ac.uk

[2]A.T.C. is with the Department of Mechanical Engineering at the University of Bristol, Bristol, UK a.conn@bristol.ac.uk

[3]J.R. is with the Department of Engineering Mathematics at the University of Bristol, Bristol, UK jonathan.rossiter@bristol.ac.uk

All underlying data to support the conclusions are provided within this paper.

Digital Object Identifier (DOI): see top of this page.

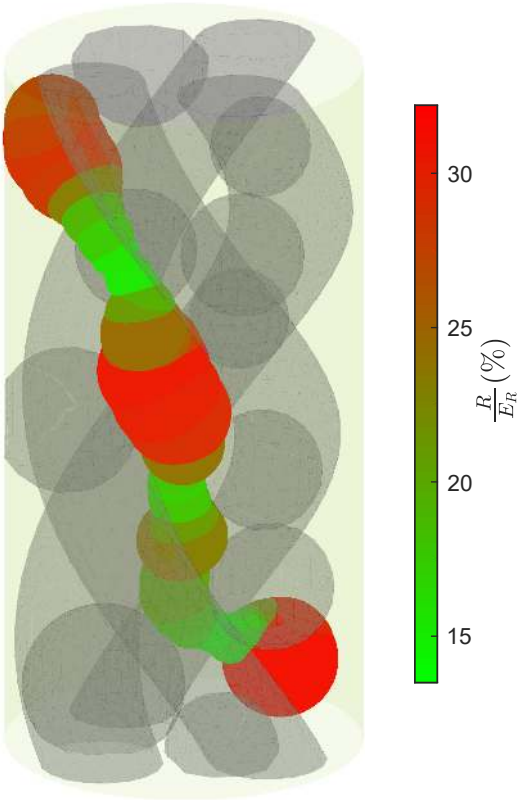


Fig. 1: Example path through congested environment of tubes and discrete spherical objects. The color scale highlights the change in localized robot diameter as a percentage of the enclosure diameter.

This work is focused on a generic navigation method for worm-like soft robots in congested pipework, cave networks or intra-organ spaces [9]. Many of these robots utilise a change in shape and size, passively or actively, to allow them to access confined environments. However, untethered devices will often need to contain pumps, a power supply and other potentially inflexible components, meaning that their range of morphing and adaptability is limited. Modes such as peristaltic locomotion that rely on gripping surroundings will also be limited by the elastic limit of inflating components. To effectively plan within these limitations, the planning method must take into account the maximum and minimum sizes that the robot can take. We present a novel method to plan a locomotion trajectory for an arbitrary morphing soft robot.

Often, robots are designed for a specific environment that

will contain a predictable set of obstacles that fall within a range of sizes and shapes. Our proposed method also serves as a useful design tool for making an informed decision on how to design a soft robot based on the intended environment.

The proposed method applies Voronoi diagrams and graph search algorithms to the navigation of mobile robots. Prior works using related approaches include using single points to represent obstacles [10] or applying a bounding box around obstacles [11], but do not consider the unique properties of morphing soft robots. In this paper we significantly expand this representation to model obstacles as a detailed point cloud through which a soft robot can navigate.

Voronoi partitioning is a fundamental geometric data structure [12] and a well-known method of representing an environment for collision avoidance and roadmap planning methods because the edges of a Voronoi diagram provide maximum clearance from a set of obstacles [13]. In this work, the vertices of the Voronoi edges will act as the centerpoints for a series of spheres that represent a worm-like robot. The application of the Voronoi planning method for mobile robots typically involves sampling the environment or using other methods to generate a point cloud. A 3D Voronoi diagram is then constructed and a graph search algorithm is applied to find the optimal path between given start and end points [14]. The cost of a path is typically defined solely by its length. In this paper, the total change in the robot's size is an additional factor used in calculating the cost, thereby including a measure of work done, which can be a better indication of the most efficient route.

In section II the Voronoi-based method of generating waypoints is introduced, followed by a description of the graph search path-planning algorithm used. In section III we discuss the effects of different environmental parameters on the path generated, and in section IV the effects of various parameters for tailoring a path are discussed. The work concludes with a discussion of the merits and limitations of the proposed method, including avenues for future work.

II. Method Overview

We first acquire a 3D point cloud of the environment. In practice, this may be achieved via external mapping technology or on-board sensors such as vision systems, LIDAR or ultrasound [15], but in this study will be assumed to be known a priori using simulated enclosures to demonstrate the application of the method presented. Before a path can be calculated, a discrete set of coordinates to be considered by the path-planning algorithm is prepared; these coordinates are referred to as waypoints. The worm-like robot is represented by a series of spheres at each waypoint. The process of generating and preparing waypoints from the point cloud is summarized by Algorithm 1.

Algorithm 1 Summarised stages of proposed algorithm

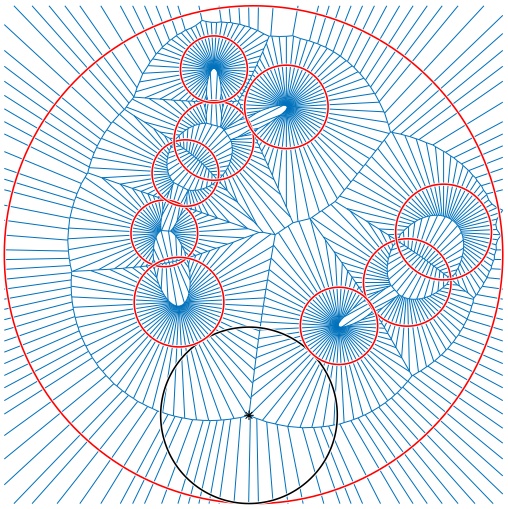
```
3D Voronoi diagram of environment point cloud.
Assign vertices of Voronoi diagram as waypoints.
Robot radius at each waypoint = distance to closest.
  environment point.
Determine adjacent pairs of waypoints from
  Delaunay triangulation.
while Any edge length < ∑ radii at paired points do
    Add bisecting waypoints to long edges
    Calculate radius at new waypoints
    Repeat Delaunay triangulation
end while
Remove edges including a radius outside of size range.
Calculate cost matrices for change in size and distance
  between adjacent pairs.
Apply graph search path-planning algorithm
```

A. Enclosure generation

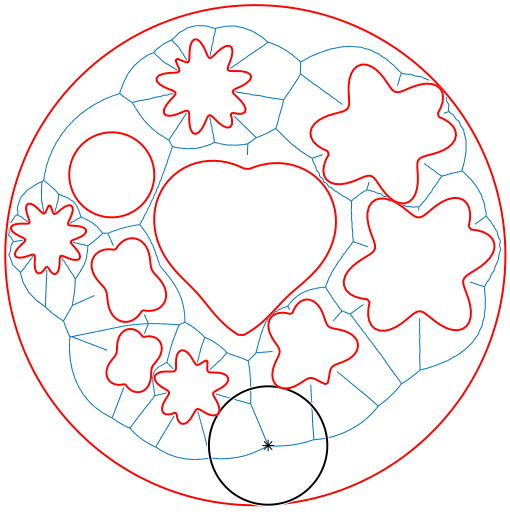
The internal structure of each cylindrical enclosure is randomly generated to account for a wide range of possible arrangements of obstacles. The nature of the enclosure is tailored by specifying parameters for the number and shape of obstacles, upper and lower bounds for their size and the separation between obstacles. Each enclosure is constructed by generating a set of random candidate obstacle coordinates within the enclosure and radii within the specified range. Candidate points are considered sequentially and added to the enclosure as obstacles if they do or do not pass outside of the enclosure, subject to the type of obstacle being modeled. Obstacles must also satisfy the separation criterion by being no closer than the lower bound, S_L, to any other obstacle and no further from the closest obstacle than the upper bound, S_U. The separation parameters, S_L and S_U, allow for a number of clusters of overlapping obstacles to be formed. For example, Fig. 2a shows two clusters, formed by specifying $S_U = \infty$ and a high S_L for the initial two obstacles that form the base for each cluster, then $S_U = 0$ and a negative S_L for the remaining obstacles, causing them to overlap. Fig. 1 shows the two types of obstacles used in this paper, namely floating spheres and continuous, twisted cylinders, although other obstacles can be used, as shown in Fig. 2b. The spheres are generated in a 3D space, whereas the cylinders are generated as a 2D cross-section that can then be rotated or otherwise manipulated to create extruded shapes throughout the length of the enclosure.

B. Waypoint generation

The set of vertices of the Voronoi diagram, or Voronoi vertices, V_V, are used as waypoints. The Voronoi diagram divides a 2D or 3D set of seed points into cells, the vertices of which are equidistant to the neighboring points. By lining the surface of an obstacle with seeds, the resulting Voronoi vertices form channels between the obstacles. Fig. 2a and Fig. 2b show the Voronoi cells and channels formed by V_V. The channels in Fig. 2b can be observed within concave features of an obstacle, creating sufficient paths to explore the enclosure thoroughly. Resembling their inspiration, worm-like soft robots usually have a circular cross section. These channels therefore form the natural position that the centreline of a soft robot would take when inflated to fill a given space. At each waypoint, the maximum radius of the robot, R, is equal to the Euclidean distance to the closest obstacle vertex, V_O, or the enclosure boundary vertex, V_E. To deliver a uniform



(a)



(b)

Fig. 2: Examples of 2D enclosures showing largest fitting circle in black with (a) two clusters of red circular obstacles formed by adjusting the separation parameters for S_L and S_U, and also showing all Voronoi cells, and (b) non-circular obstacles, showing only Voronoi channels.

distribution of Voronoi points around obstacles, V_E and V_O should also be uniformly distributed across their respective surfaces. To achieve this, spherical obstacles are represented as icospheres with a number of vertices proportional to their radius, and continuous obstacles are given a similarly uniform surface mesh.

C. Graph search path-planning algorithm

From V_V, paths are defined in 3D space using Dijkstra's shortest path algorithm [16]. Dijkstra's algorithm was chosen as it is easily implemented, although other node-based optimal algorithms such as A* could also be used [14]. Dijkstra's algorithm is a breadth-first search method that finds the optimal path - the path with the lowest cost - by expanding from the start point to adjacent vertices and calculating the cost to reach each vertex until the end point has been reached. In general, a graph is denoted $G(V, K)$, where V is a set of vertices and K is a set of edges, uv, connecting adjacent vertices $u \in V$ and $v \in V$. The edge list, K, can be generated from a Delaunay triangulation of V, which creates a triangular mesh between all adjacent vertices [17].

Generating the edge list from V_V alone could result in paths that pass directly through obstacles, therefore V_E and V_O are also included. The Delaunay triangulation of V_A, where $V_A = \{V_V \cup V_E \cup V_O\}$, produces the initial edge list, K. K is modified so that any edge including a vertex from V_E or V_O is excluded, giving adjacent pairs for only V_V that do not pass through any obstacles. A potential limitation in using a Voronoi diagram to generate waypoints is that the density of V_V is driven by the density of V_O and V_E. Having too low a density of V_V can result in an edge for which the distance between the two points $d(u, v) > R(u) + R(v)$ and hence point v could be unreachable if there is a closure between the two points that is smaller than the minimum size of the robot. Conversely having too high a density leads to unnecessarily expensive computation. To ensure that the robot can 'reach' an adjacent vertex, regardless of the density of V_O and V_E, bisecting points between u and v can be added to V_V when the distance between vertices $d(u, v) > R(u)+R(v)$, and both $R(u)$ and $R(v) > R_{min}$, the minimum size of the robot. R is extended to include radii for any new points and the Delaunay triangulation is repeated until no new points are added. Note that this does not negate the effect of a low input-point density on error in R (see discussion section). After any points have been added, a final modification is made that excludes edges that connect to a point with R outside the defined size range for the robot, so that $R = [R_{min}, R_{max}]$.

In Dijkstra's algorithm, the cost to move between adjacent verticies, u and v, is given by the cost matrix $C(u, v)$, where $C \in \mathbb{R}^{n \times n}$ and $n = |V_V|$. In this paper, the total cost matrix (1) is calculated by the weighted sum of the distance between vertices and the change in radius. A cost matrix is calculated for each factor then normalized to the interval $[0, 1]$ so that they influence the path proportionately.

$$C = w_d C_d + w_r C_r \tag{1}$$

Where C_r is the radius cost matrix and C_d is the distance cost matrix. w_d is the weight of distance and w_r is the weight of change in R. The separate costs of a path starting at v_1 and finishing at v_k are defined by (2) and (3).

$$r(v_1, v_k) = \sum_{i=2}^{k} C_r(v_{i-1}, v_i) \tag{2}$$

$$d(v_1, v_k) = \sum_{i=2}^{k} C_d(v_{i-1}, v_i) \tag{3}$$

Where r and d are the radius cost and distance cost of a path, respectively. The choice of end points depends on the goal of the robot. If its purpose is to explore a space for maximum coverage or to progress in a general direction, multiple possible end points can be selected and the lowest cost path chosen from the multiple paths generated. Alternatively, there may be a specific location that the robot should reach, in which case

only that single point would be chosen. For tests with a single start and end point, these are waypoints with R closest to the midpoint of R_{min} and R_{max}, as this would mimic where the robot would most likely be if it was part way through a larger environment. This selection can have a significant impact on the success of generating a path and should be tailored for any given robot. For tests with multiple possible end points, the method of selecting a start point is unchanged, but the end points are randomly selected from vertices towards the end of the enclosure. In these tests, 100 possible end points are chosen as it was found that, beyond this number, paths would tend to wind around the vertices at the end of the enclosure rather than finding new routes through the bulk of the enclosure.

III. Influence of Enclosure Generation Parameters

The influence of the enclosure generation parameters are explored individually, while keeping other parameters constant. Fig. 3 shows trends in the distribution of robot radius, R, as a percentage of enclosure radius, E_R. Increasing the size or quantity of obstacles leads to more waypoints being generated, therefore the frequency of each value of R is shown as a percentage of $|R|$ for the sake of comparison. The effect of having either more obstacles (Fig. 3a) or larger obstacles (Fig. 3b) is to increase the frequency of smaller values of R and to reduce the overall range of R. This is to be expected as a more cluttered environment will only permit the passing of a smaller robot.

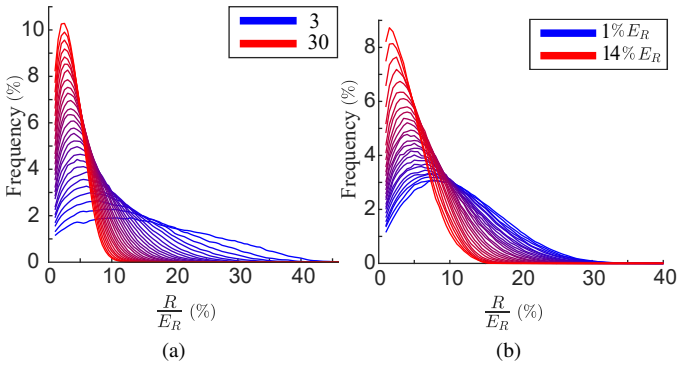


Fig. 3: Histograms of influence of parameter size on distribution of R as a percentage of E_R: (a) number of obstacles and (b) radius of spherical obstacles relative to E_R.

IV. Influence of Path Selection Constraints

As R_{min} and R_{max} are increased and decreased respectively, the overall trend in Fig. 4 shows that a narrower range of sizes leads to fewer successful paths, and the paths that are possible have a higher distance cost. For this comparison, a single end point is selected and generation parameters for enclosures with spherical obstacles and continuous obstacles are adjusted to have a similar total volume of obstacles. Comparison of Fig. 4 (b) and (d) shows that the overall trend is the same for both obstacle types, but the size limits have a sudden cut-off point for the success rate with continuous

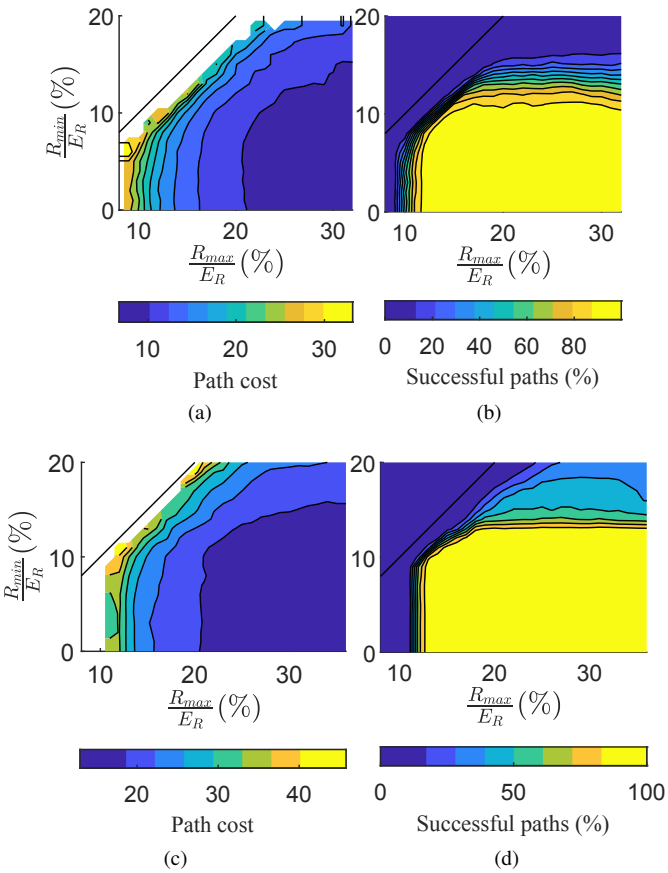


Fig. 4: The effect of R_{min} and R_{max} on the (a) cost in environment with spherical obstacles, (b) percentage of successful paths with spherical obstacles, (c) cost with continuous obstacles, and (d) percentage of successful paths with continuous obstacles.

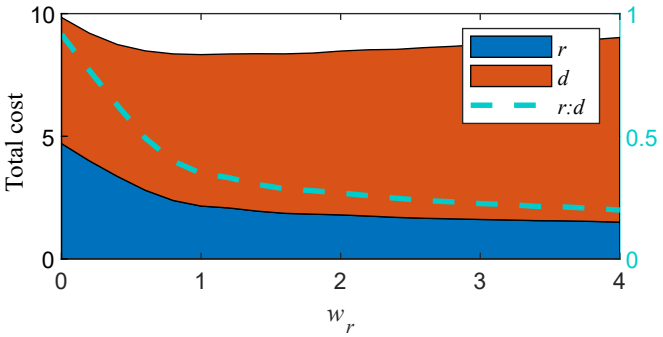


Fig. 5: The effect of w_r on total cost contributions from the distance cost, d, and the radius cost, r, for enclosures similar to Fig. 6c, with w_d = 1. The ratio r:d is also shown for clarity.

obstacles. This pattern may be the result of a smoother and less chaotic environment than one with spheres, which provides more opportunities to weave between the obstacles. Fig. 5 shows that radius cost of a path, r (2), is inversely proportional to radius weight, w_r, for the chosen path.

Examples of the influence on the shape of a path by radius weight, w_r, distance weight, w_d, and size limits of the robot,

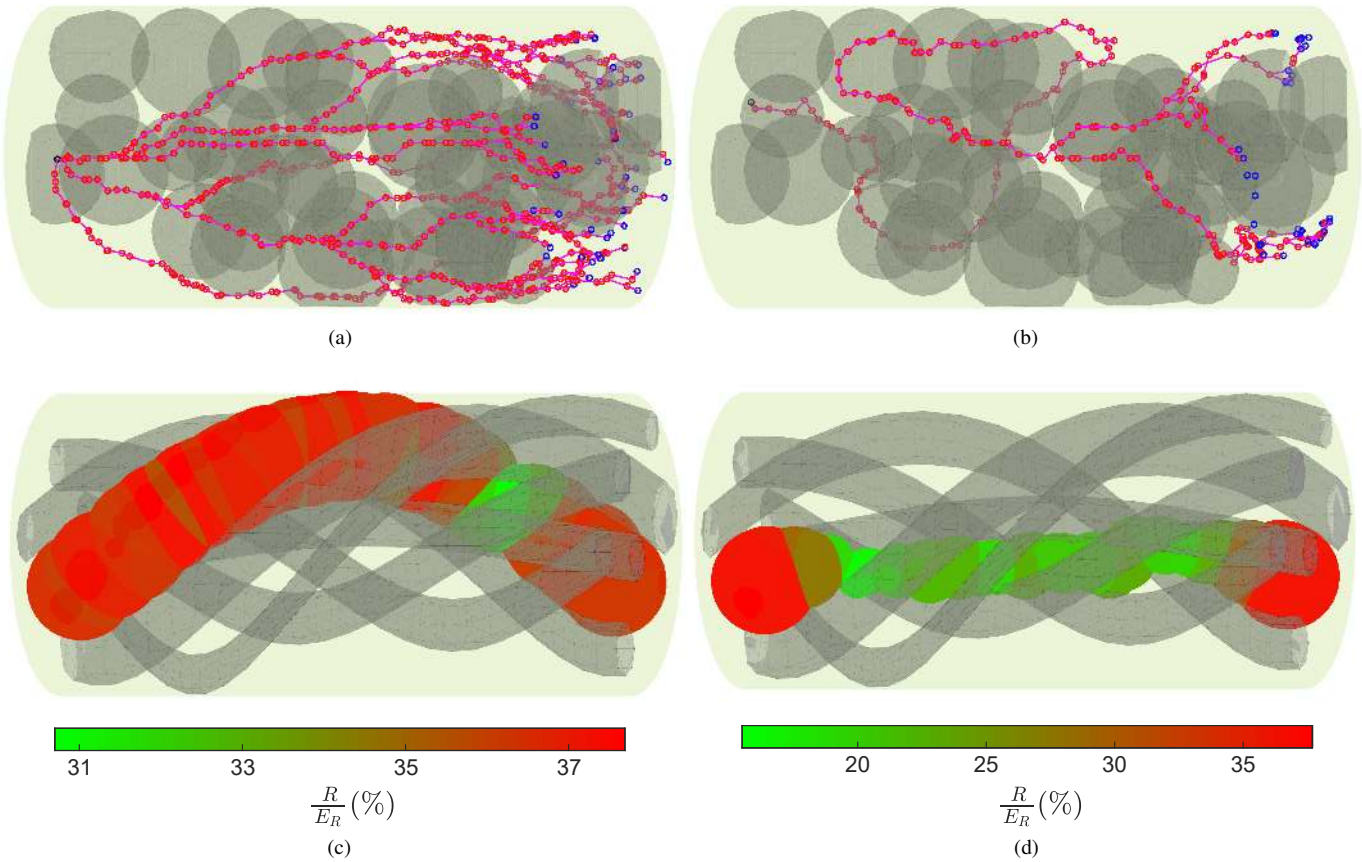


Fig. 6: Influence of size range on number of available paths with (a) open size range and (b) narrow size range. The start point is shown as a black marker and the possible end points are shown as blue markers. Influence of cost ratio on cost of path with (c) $w_r = 1$ and $w_d = 0$, and (d) $w_r = 0$ and $w_d = 1$. The color scale highlights the change in localized robot diameter as a percentage of the enclosure diameter.

R_{min} and R_{max}, are shown in Fig. 6, where (a) and (b) use multiple end points, showing all possible paths, and (c) and (d) use the same start and single end point. It can be observed that the open size range in (a) leads to paths branching off into many more direct paths, whereas the narrow size range in (b) is limited to a single, winding route through the majority of the enclosure and the end points within the size range are more clustered together. Fig. 6 (c) and (d) show that w_r and w_d are effective in tailoring a path with a bias for minimizing change in radius or the length of a path.

V. Discussion and Future Work

The Voronoi method employed in this application is relatively robust in that, even with a low input-point density, the Voronoi channels do not significantly change position, only resolution. However, the error in robot size from calculating R based on the single closest point of $V_E \cup V_O$ is dependent on the input-point density. The error in R is proportional to the distance between two neighboring points of V_O relative to the curvature of the obstacle. In the case of a convex obstacle, R will be overestimated and overlap with the obstacle, whereas a concave shape will cause R to be underestimated. As soft robots commonly rely on pressure sensors to detect contact [18], a small error in R will have little impact in practical applications. However, when using the proposed method to guide robot design, it is better to underestimate the size of a gap than to overestimate it.

Since the number of waypoints, $|V_V|$, is proportional to $|V_E \cup V_O|$, in an empty section of an enclosure the lack of waypoints would create a path that follows the centerline. This is a limitation of the method as, in this case, $R = E_R$, if $E_R > R_{max}$ then the algorithm would not be able to return a suitable route. However, a more natural trajectory may be used by the robot that involves moving along the floor of the empty enclosure. A solution for making the method more robust in these cases would be to populate spaces devoid of any V_O or V_V with an array of dummy waypoints. Open spaces such as these could also have a cost associated with gravitational potential to make the floor of an enclosure more appealing. Other costs such as the difference between R and the robot's neutral size can also be easily added, to better determine an efficient route in the case that there is a relationship between size and velocity or work done. A constraint for the maximum angle between points or an angle cost can be added, however this would require a different approach using, for example, a 3D cost matrix. Another avenue of future work in more

varying environments is to implement dynamically changing cost weights to suit the local environment.

A limitation of this algorithm is found when planning to cross a gap between features of the environment. The method presented assumes that contact with the environment is maintained at each waypoint along the path, whereas in reality a large portion of the robot's body could be supported by fewer contact points. The algorithm can be altered to include the length of the robot and a tally of the number of waypoints for which $R > R_{max}$, that is, where the robot is not in contact with the environment. In this way gaps can be crossed. When the tally to reach a particular vertex exceeds a defined limit, the cost would default to infinity, thus marking it as unreachable. This modification also makes the algorithm more robust when applied to robots with a higher elastic modulus by allowing for a narrower range of expansion without limiting the number of available paths, as illustrated in Fig. 6b.

The examples in this paper model a relatively small section of a closed environment with a complete dataset, however this method is also suitable for a localized set of points without a finite border. In this case, a meshed convex hull can be generated around local data points to prevent a path being generated that would skirt the point cloud through unknown space. In practice, there is likely to be environmental uncertainty due to noise or lack of sensory penetration, subject to the type of 3D scanner(s) being used. To reduce this uncertainty, depending on the specific environment and robot, it could be possible to have a coarse route planned ahead with data collected from sensors outside the environment. On-board sensors would then deliver a more refined localized map and route in real-time. A goal for future works is to explore performance when using an incomplete dataset and to explore how environmental uncertainty can be reduced.

Regarding time complexity, a large portion of computation time is taken up by Dijkstra's algorithm, which runs in time $\mathcal{O}(|K|\log|V_V|)$ [19]. The remaining time is largely dependent on R_{min}, as this will strongly influence how many times the bisecting procedure repeats. When R_{min} is set to be equal to the lowest value of R along the calculated path, the time excluding Dijkstra's algorithm can be summarized as being time $\mathcal{O}(|V_V|)$.

Here we use a series of spheres to represent sections of the soft worm-like robot. This does not account for passive deformation as the robot squeezes against its environment. The robots illustrated here are also assumed to be able to change volume. Another interesting constraint for this method would be to account for having a fixed volume, especially in combination with a tally of the number of secure points. A goal of future works is to perform proof-of-concept experiments to determine how material properties and a fixed volume may influence the optimal path.

VI. Conclusion

In this work, a novel method of path planning for soft robots in congested environments is presented, utilizing 3D Voronoi diagrams and graph search algorithms. By employing a cost function that includes the required change in a robot's diameter in addition to the distance between two waypoints, the most efficient route can be calculated. The method shown is suitable for congested sections of an enclosure and has the potential to be easily modified to account for more diverse environments such as crossing large gaps and unobstructed sections without the need to employ a different planning method. This work provides the foundation for efficient path planning for soft morphing robots and their wider application in industry, healthcare and the exploration of tight spaces.

References

[1] C. Lee, M. Kim, Y. J. Kim, N. Hong, S. Ryu, H. J. Kim, and S. Kim, "Soft robot review," *International Journal of Control, Automation and Systems*, vol. 15, no. 1, pp. 3–15, 2017.

[2] B. Trimmer, "A Journal of Soft Robotics: Why Now?" *Soft Robotics*, vol. 1, no. 1, pp. 1–4, 2014.

[3] K. M. Digumarti, A. T. Conn, and J. Rossiter, "Euglenoid-Inspired Giant Shape Change for Highly Deformable Soft Robots," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2302–2307, 10 2017.

[4] A. S. Boxerbaum, K. M. Shaw, H. J. Chiel, and R. D. Quinn, "Continuous wave peristaltic motion in a robot," *International Journal of Robotics Research*, vol. 31, no. 3, pp. 302–318, 2012.

[5] B. Zhang, Y. Fan, P. Yang, T. Cao, and H. Liao, "Worm-Like Soft Robot for Complicated Tubular Environments," *Soft Robotics*, vol. 6, no. 3, pp. 399–413, 2019.

[6] H. Abidi, G. Gerboni, M. Brancadoro, J. Fras, A. Diodato, M. Cianchetti, H. Wurdemann, K. Althoefer, and A. Menciassi, "Highly dexterous 2-module soft robot for intra-organ navigation in minimally invasive surgery," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 14, no. 1, p. e1875, 2018.

[7] E. W. Hawkes, L. H. Blumenschein, J. D. Greer, and A. M. Okamura, "A soft robot that navigates its environment through growth," *Science Robotics*, vol. 2, no. 8, p. eaan3028, 2017.

[8] J. H. Boyle, S. Johnson, and A. A. Dehghani-Sanij, "Adaptive undulatory locomotion of a C. elegans inspired robot," *IEEE/ASME Transactions on Mechatronics*, vol. 18, no. 2, pp. 439–448, 2013.

[9] T. Deng, H. Wang, W. Chen, X. Wang, and R. Pfeifer, "Development of a new cable-driven soft robot for cardiac ablation," *2013 IEEE International Conference on Robotics and Biomimetics, ROBIO 2013*, pp. 728–733, 2013.

[10] A. Singh, Anshul, C. Gong, and H. Choset, "Modelling and Path Planning of Snake Robot in cluttered environment," in *2018 International Conference on Reconfigurable Mechanisms and Robots, ReMAR 2018 - Proceedings*, 2018, pp. 1–6.

[11] M. Candeloro, A. M. Lekkas, J. Hegde, and A. J. Sorensen, "A 3D dynamic Voronoi diagram-based path-planning system for UUVs," in *OCEANS 2016 MTS/IEEE Monterey*, 2016, pp. 1–8.

[12] F. Aurenhammer, "Voronoi diagrams—a survey of a fundamental geometric data structure," *ACM Comput. Surv.*, vol. 23, no. 3, p. 345–405, Sep. 1991. [Online]. Available: https://doi.org/10.1145/116873.116880

[13] P. Bhattacharya and M. L. Gavrilova, "Voronoi diagram in optimal path planning," in *4th International Symposium on Voronoi Diagrams in Science and Engineering (ISVD 2007)*, 2007, pp. 38–47.

[14] L. Yang, J. Qi, D. Song, J. Xiao, J. Han, and Y. Xia, "Survey of Robot 3D Path Planning Algorithms," *Journal of Control Science and Engineering*, vol. 2016, pp. 1–22, 2016.

[15] H. Cho, B. Kim, and S. Yu, "AUV-Based Underwater 3-D Point Cloud Generation Using Acoustic Lens-Based Multibeam Sonar," *IEEE Journal of Oceanic Engineering*, vol. 43, no. 4, pp. 856–872, 10 2018.

[16] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathmatik*, no. 1, pp. 269–271, 1959.

[17] D. T. Lee and B. J. Schachter, "Two algorithms for constructing a Delaunay triangulation," *International Journal of Computer & Information Sciences*, vol. 9, no. 3, pp. 219–242, jun 1980. [Online]. Available: https://link.springer.com/article/10.1007/BF00977785

[18] A. A. Calderón, J. C. Ugalde, L. Chang, J. C. Zagal, and N. O. Pérez-Arancibia, "An earthworm-inspired soft robot with perceptive artificial skin," *Bioinspiration & Biomimetics*, vol. 14, no. 5, 8 2019.

[19] M. Barbehenn, "A note on the complexity of dijkstra's algorithm for graphs with weighted vertices," *IEEE Transactions on Computers*, vol. 47, no. 2, pp. 263–, 1998.