Unions of Balls and Configuration Space

Yinan Zhang
Department of Computer Science
Dartmouth College
Hanover, New Hampshire 03755, US

Devin Balkcom
Department of Computer Science
Dartmouth College
Hanover, New Hampshire 03755, US

August 18, 2014

Abstract

Covering both the configuration space, or *C-space*, and represent the right topology is hard to achieve at the same time with traditional sampling-based motion planning algorithms, such as Probabilistic Roadmap and Rapidly-Exploring Random Tree. In this paper, we proposed a method that is able to cover near optimal path and study the shape and topology of configuration space. Our method samples a serious of balls to cover the space and use the dual shape of the union of balls to re-construct space. We analysis the coverage of the space and show how these balls and their dual shape can be used to generate a graph that maintains the topology of *C-space*

1 Introduction

Planning a collision-free motion for movable object could be very challenge, especially in high dimensional space, because of the lack of knowledge to understand the shape and top of configuration space. Meanwhile, motion planning has extensive applicationsto areas like augmented reality [1], computer-aided design [2] and bioinformatics [3]. More knowledge on *C-space* will be benefit to these research.

Although we have seen some study in low dimensional space trying to characterize the topology of *C-space* [4] [5], these methods suffer dimension curse in high dimension space.

Sampling-based method for motion planning appear to be very successful [6]. These methods are usually proved to be *probabilistically complete*, thus guarantee to find a path if it exists. Medial Axis sampling [10] [7] attracted researchers attention for their property of characterizing the topology of *C-space* and safe clearance to obstacles, however, no structures are made of these samples to properly represent the space.

Trying to answer the question "does a collision free path exist", Zoe McCarthy uses samples to build an α-shape to better understand samples and space captured, which inspires us to explore more with α-shape.

Approximating the "shape" of a given point set in a space is a frequently occurred question in many research areas, which is the intuitive of α-shape. A relative notion is space-filling diagram which is used in chemistry and biology to represent molecular structures. [13].

In this paper, we will firstly introduce sampling algorithms to cover the *C-space*, medial axis sampling methods are applied to reduce the number of samples and capture the topology of the space. We then use these samples, or balls, to construct the weighted α-shape. After simplifying the α-shape, we will still have a simple graph, which can be used for both motion planning and topology study. We at each step, we will demonstrate algorithm results of a 2D configuration space. Some properties of our algorithm are discussed in section 4.

2 Related Work

In this section, we first discuss the basics of sampling-based motion planning and the method of sampling on medial axis. With the information of positions and their clearance in c-space, we have samples as spheres. We then discuss the dual shapes of the union of balls which plays an important role in understanding the shape of c-space and its topology. Some works on graph theory and topology will finally be introduced.

A *Sampling-based Motion Planning*

A robot is a movable object whose state can be described by n parameters, or *degrees of freedom* ($DOFs$). A point $< x_1, x_2, ..., x_n >$ in a n-dimensional space uniquely defines the configuration of a robot. Such space is called *"Configuration Space"* or *"C-space"* (C). The subset of all feasible configurations is called the *free space* (C_{free}), while $C_{obst} = C \setminus C_{free}$ is the union of all infeasible configurations. [7] Path planning will generally be viewed as a search in a metric space X for a continuous path from an initial state x_{init} to a goal region $X_{goal} \subset X$. For a standard problem, $X = C$. [8] What needs to be pointed out is that *C-space* is very different from workspace which is usually formed by polygons. The shape of *C-space* is usually unknown to us, unless the robot is a point.

Sampling-based algorithms have been very successful and have seen many applications in industry for planning in high dimension space. Probablistic Roadmap [9] and Rapidly-Exploring Random Tree [8] are two outstanding methods. The basic idea under these methods is to randomly sample feasible configurations in *C-space* and connect nearby valid samples as edges to construct a graph or tree structure, which careless about the dimension problem. By connecting start and goal configurations to the roadmap, a graph search, e.g., A*, can be performed to extract a near optimal solution path.

One disadvantage of these methods is that we can't study the topological structure of C-space using only the data structures constructed.

B *Medial Axis Sampling*

A simpler method that maintains the topology of *C-space* but doesn't cover near optimal path is medial axis sampling. Medial Axis is also called Generallized Voronoi Diagram. The medial axis $MA(F)$ of free space F is a strong deformation retract of F, which means F can be continuously deformed onto $MA(F)$ while maintaining its topological structure. However, medial axis is hard to compute explicitly.

In Figure [6], sec. 5.6.3, Steven LaValle defined Medial Axis as "Let (X, ρ) be a metric space. Let a *(*maximal ball) be a ball $B(x, r) \subseteq X$ such that no other ball can be a proper subset the centers of all maximal balls trace out a one-dimensional set of points referred to as the *medial axis*".

Mapping a pair of points A and B in free space onto $MA(F)$, there is a path between A and B if and only if there is a path between their images on $MA(F)$. Such path has a very nice property of the largest clearance from obstacles.

Steven A. Wilmart [10] sampled on the medial axis of free space to build up a probabilistic roadmap planner. He also showed that "it is possible to efficiently retract a configuration, free or otherwise, onto the medial axis of the free space without having to compute the medial axis explicitly".

The basic idea of the algorithm proposed in [10] is: given a point A and its nearest point on the boundary of obstacles A', a ray start from A with direction $\overrightarrow{A'A}$ will always intersect with $MA(F)$.

This method results in sampling more nodes in narrow corridors, which

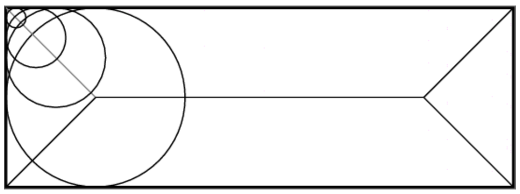

Figure 1: Sampling on the Medial Axis of a rectangle space.[6]

is beneficial for improving performance on problems requiring traversal of such corridors.

However, such property makes the sampling method biased heavily towards certain portions of the medial axis. These biased property tend to ignore more critical parts of the space: large open free space. Sampling uniformly on the medial axis is sometimes more helpful.

Hsin-Yi (Cindy) Yeh [7] solved this problem by putting some random "sticks" with random directions in free space. Any surfaces will intersect uniformly with these sticks, thus searching on sticks to find points that intersect with medial axis will give nodes uniformly distributed on the medial axis.

Figure 1 visualizes the shape of medial axis of a rectangle.

C *Dual Shapes of Unions of Balls*

Approximating the shape of an unknown area to study its properties is a very commonly used method. We have seen many such ideas in the study of biogeometry. As for robotics, a few people began to notice the achievements from biogeometry study and tried to apply similar methods to study the shape of *C-space*. [13]

Zoe McCarthy [16] used samples generated by PRM to form the α-shape of *C-space*. She then proved there doesn't exist a path between a pair of configurations by proving they are in two different disconnected components of the shape. She declared it was the first time α-shape is being used in motion planning.

Define a *simplicial complex* D: a collection of simplices such that if $\triangle_T \subseteq D$ ($\triangle_T$ is the convex hull of T), then if $U \subseteq T$, we have $\triangle_U \subseteq D$ and also that the intersection of two simplices in D is either empty or another simplex in D. The α-shape of a set of points S is a generalization of the convex hull of those points. [16]. H. Edelsbrunner first introduced them in [14]. By choosing different value of the one parameter α, we can have a family of shapes with the same set of points. An example of α-shape is in

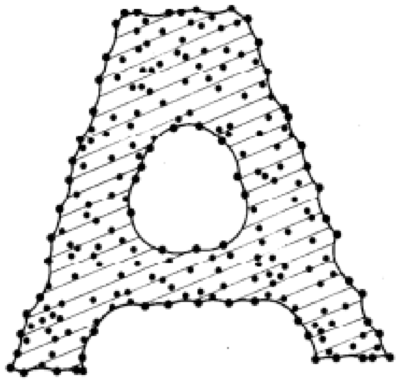

Figure 2: An example of α-shape. [14]

Figure 2.

Choosing the best α value could be a bothering work. Assuming weighted points in the point set, by using a generalization of the Delaunay Triangulation, the Regular Triangulation, we can construct another shape using weights as flexible α's. Points with weights can be considered as balls. H. Edelsbrunner gave us this idea in [12]. Such shapes are also called Dual Shapes or Dual complex of unions of balls. The weighted α-shape captures the topology of the union of balls with different radii. Figure 3 shows an example of such weighted α-shape.

D *Betti Numbers*

In algebraic topology, the Betti Numbers are used to distinguish topological spaces based on the connectivity of n-dimensional simplicial complexes. Informally, the *n-th* Betti number (β_n) represents the rank of the *n-th* homology group, which tells us the maximum number of cuts that can be mad before diving a surface into two pieces. [17]

An example of Betti numbers will be a torus. Figure 4

The first betti number of a graph is very easy to compute. Given a graph G, with n vertices, m edges and k components. It's 1st betti number is:

$\beta_1 = m - n + k$

This can be proved straightforwardly by mathematical induction on the number of edges. A new edge either increments the number of 1-cycles or decrements the number of connected components. [17]

H. Edelsbrunner introduced a way to compute $n - th$ Betti number in [18], where he uses gaussian ellimination to compute the betti numbers with

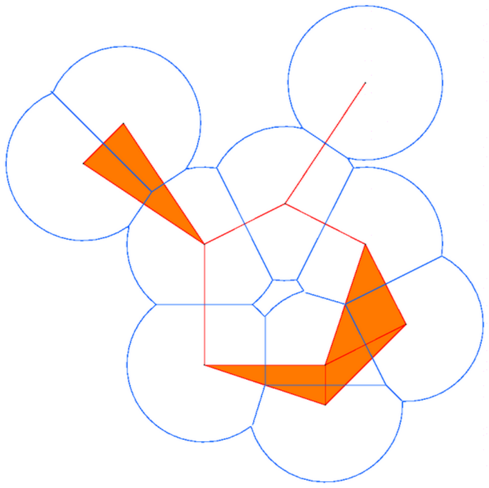

Figure 3: An example of weighted α-shape for a union of balls. [12]

adjacency matrix.

Vanessa Robins [19] provided more detailed explains and proofs on computing betti numbers. What's more, she put more thoughts on computing betti numbers for alpha shapes.

3 Our Method

Traditionally, sampling based algorithms will build a graph for configuration space. These kind of graphs are proved to be able to find near optimal path, but can hardly be used to analysis the topology of paths. We want to to build a stronger representation of c-space that is both able to cover optimal path and study the topological structure of c-space. In this section, we are trying to solve the problem in 2D and answer the question "Is two paths in the same homotopy class" as a start point of using our method to study the topology of *C-space*. We will first give the outline of the algorithm, and explain each step in detail.

A *The Algorithm outline*

Before starting introducing our algorithm, we firstly clarify some preliminary tools we can have: We are sampling in (X, ρ) metric space. Let x be a configuration in X(usually *C-space*), $\rho(x)$ gives the clearance of configuration x. By *"clearance"*, we mean its (estimated) distance to the nearest

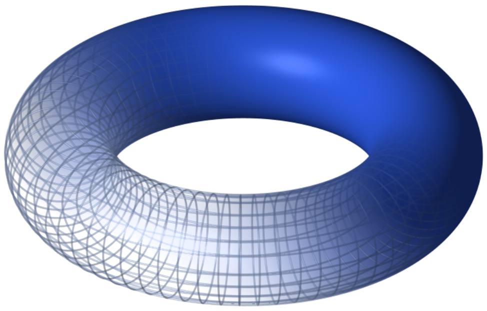

Figure 4: The torus has one connected component(β_0), two circular holes (β_1,the one in the center and the one in the middle of the "donut"), and one two-dimensional void (β_2, the inside of the "donut").[17]

obstacles. Ball $B(x, \rho(x))$ is referred as a "*sample*" in the rest of the section. We make no assumption of the shape of obstacles regions in *C-space*.

The algorithm is shown in Figure 5. It has four phases. The first phase is to generate some random configurations in *C-space*, like all other sampling-based planning algorithms do, and keep only those feasible ones. Every feasible configuration will be kept until the third phase. ($randomConfig$ keep unchanged.) In the second phase, we will sample balls centered on medial axis first and then try to cover the left area of C_{free}. Samples on the medial axis $MA(C_{free})$ will be used in phase three to construct the dual shape that captures the topology of C_{free}. After simplifying the dual shape, we will have a roadmap while maintaining the topological structure. Phase four break the topology roadmap into pieces such that no two piece can be merged without forming a loop.

We will now explain each phase in detail.

B *Sampling in C-space*

As pointed out in Figure 5, this phase has two parts: Sampling on medial axis, and sampling the rest of free space.

Sampling on medial axis is well studied in [10] and [7]. Besides the positions of configurations, we also keep the clearance of each configuration, thus have some balls, or samples. We also require that no new ball, when sampling, is allowed to be centered inside any existing one. For details of the algorithms, please refer to these two papers. One thing worth our notice is that using only algorithm mentioned in [10], we can get good result

```
1: function WeNeedAGoodName(C-space)
       // Phase One
2:     $randomConfigs \leftarrow SampleRandomConfigurations(\textit{C-space}, number)$
       // Phase Two
3:     $maSamples \leftarrow SampleOnMedialAxis(randomConfigs)$
4:     $regSamples \leftarrow SampleLeftArea(maSamples, randomConfigs)$
       // Phase Three
5:     $dualShape \leftarrow WeightedAlphaShape(somesmaples)$
6:     $topologyRoadmap \leftarrow Retract(dualShape)$
       // Phase Four
7:     $brokenRoads \leftarrow BreakGraph(topologyRoadmap)$

8: end function
```

Figure 5: Algorithm Outline

in narrow corridor but are more likely to ignore large open area. While using only [7] method, we can cover large open area well, but more likely to produce disconnected balls in narrow corridor. Therefore, running both algorithm is highly recommended.

The reason we sample on medial axis first is that 1. these balls are maximal ones as the definition of medial axis suggests, and 2. medial axis $MA(C_{free})$ is a SDR of *C-space.* [11]

Figure 6 describes how we sample area not yet covered by balls centered on medial axis. The basic idea behind the algorithm is to keep sampling new balls on the boundary of existing ones by "pushing" random configurations inside existing balls to the boundary of them and determine if they are in any other balls. We restrict that all balls to be larger than a minimal radius ball, thus void getting a lot of very small samples.

Getting configurations inside a ball doesn't have to be $O(n)$ costly. It can be done using KNN algorithm, which is more efficient than iterating everything. Since we generated random configuration n times, not all are feasible, in a 2D space with area A. We expect to have n/A configurations in an unit area. A ball with radius r has area πr^2, approximately $k = \frac{n\pi r^2}{A}$ configurations are inside the ball. We then project these (uniformly distributed) configurations to the boundary of the ball. Actually, choosing a larger k could be better: we virtually extend the radius of our discs, which results in a smaller density in arcs that are very close to obstacles than those far from obstacles. Samples got in this part is called *regular samples.*

```
function SampleLeftArea( maSamples, randomConfigs )
    $regSamples \leftarrow EmptySet$                          ▷ Set of balls not on MA
    $newBall \leftarrow$ True                                 ▷ If we can get new balls
    $lastRoundBalls \leftarrow maSamples$
    while $newBall$ do
        $newBall \leftarrow False$
        $bndConfigs \leftarrow EmptySet$              ▷ A set of ball boundary configs
        for $ball \in lastRoundBalls$ do
            $innerConfigs \leftarrow KNearest(ball, randomConfigs)$;   ▷ KNN
            $bndConfigs.append(getBoundaryConfigs(ball, innerConfigs))$
        end for
        $lastRoundBalls \leftarrow EmptySet$
        for $config \in bndConfigs$ do
            if $config$ not inside any existing ball then        ▷ $O(n^2)$ time
                $ball \leftarrow getNewBall(config)$                 ▷ get a new ball
                if $ball.radius \geq min\_radius$ then
                    $newBall \leftarrow True$
                    $lastRoundBalls.append(ball)$
                    $regSamples.append(ball)$;
                end if
            end if
        end for
    end while
    return $regSamples$
end function
```

Figure 6: Algorithm to sample left free area

Figure 7 is a result of this phase running in a rectangle world with 3 obstacles in the middle. Gray balls are centered on medial axis, while green ones are not. Red points are centers.

C *Constructing Topology Roadmap of C-space*

After getting these samples, we can now start to build a structure that captures the topology of the space.

This phase, like the previous one, has two parts: 1. Construct the dual shape of medial axis samples got from phase one. 2. Simplify the dual shape to a simple graph.

In the first part, constructing the dual shape using only the medial axis

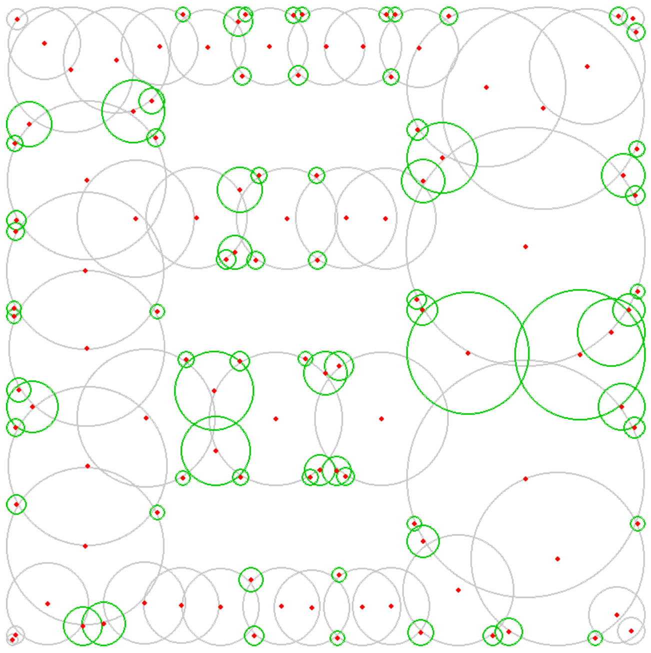

Figure 7: Samples got by the algorithm

sample could break the topology of medial axis. The reason is shown in Figure 8: three balls are all centered on medial axis, but their dual shape will be an empty triangle, which introduces a "hole" in that area. To solve this problem we can try to sample a ball centered at the the radical center after getting an empty triangle, and rebuild the dual shape for these four balls. [15] Proving the solution is very simple: if there is no obstacles in that area, the newly sampled ball will cover the space, if there is obstacle, the ball will either not be sampled or not connect three of the balls.

With these balls, we then compute their dual shape, which is formed by some 1-simplices and 2-simplices in 2D. The runtime of computing dual shape will be $O(n^2)$ which is the worst case when doing regular triangulation of weighted points. Although using all balls will avoid problem as in Figure 8, we are making n much larger, which results in a bad runtime of this step. (However, constructing the dual shape of all balls will be helpful to study the shape of *C-space*)

One might argue that we can simply get configurations on medial and connect them as a roadmap. However, this roadmap doesn't maintain the topology of *C-space*. At some corners, we will still get triangles which don't

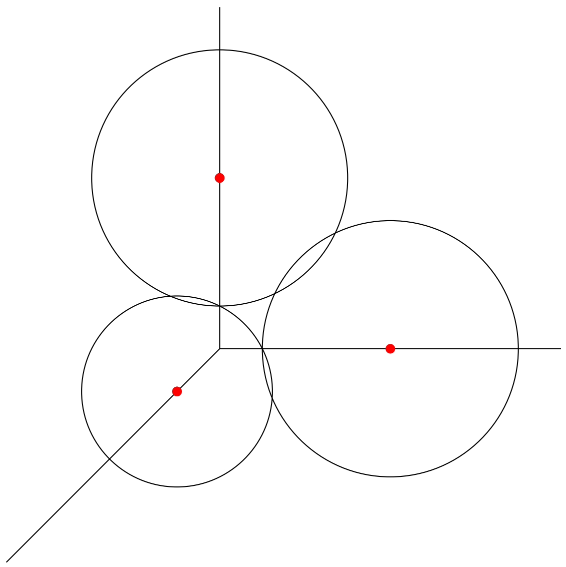

Figure 8:
An example that medial axis samples break the topology structure of the medial axis.

have information if there are obstacles inside the triangle. By building the dual shape, this will no longer be a problem.

The second part of this phase can be stated as: retract all connected 2-simplices into one point, reconnect these retracted points with other ones using their previous connection.

D *Breaking A Graph into Loop-free Parts*

After getting the topology roadmap, we technically already have a planner, but still we can simplify it. We now discuss breaking a graph into several components, such that no two components can merge into a new graph without forming a loop. By doing this, a graph is even simplified. Knowing which components are the start and goal configurations in, we can find a path connecting these components immediately. This technique can be a great improve when $C - space$ is very large and traditional PRM or RRT methods will take a long time to solve.

Let g_1 and g_2 be two graphs without loops. We say g_1 and g_2 to be mergeable if: 1. they share at least one common node, 2. the new graph formed by these two graph has no loop.

Determine if a simple graph has a loop is very easy using the first Betti

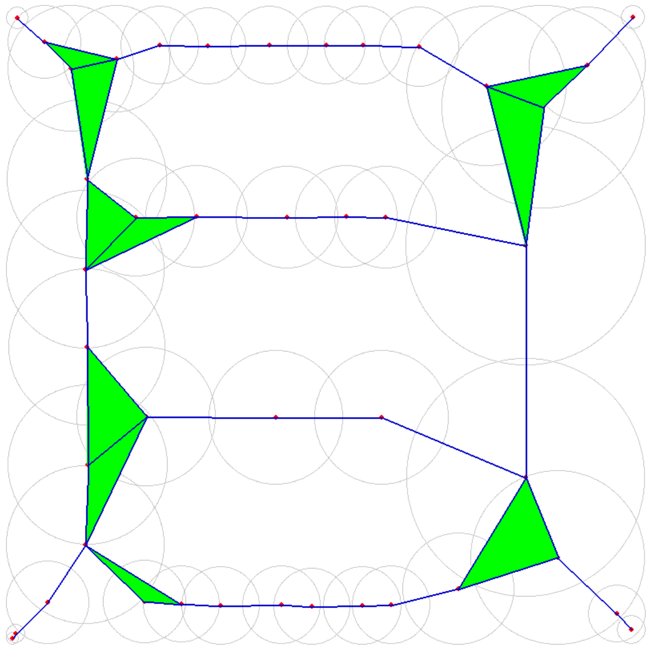

Figure 9:
Dual shape of balls generated in phase 1 (centered on medial axis).

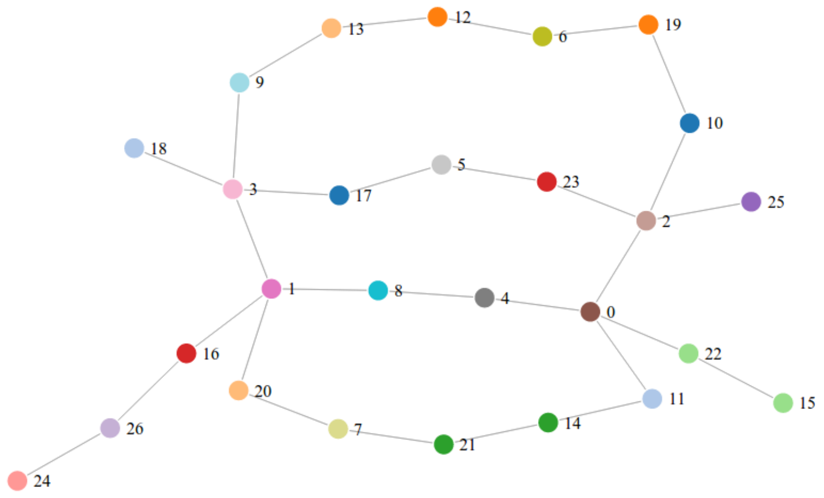

Figure 10:
Simplified graph of the dual shape in Figure 9.

```
function BreakGraph( graph )
    $components \leftarrow EmptySet$
    for $edge \in graph.edges$ do
        components.append( new Graph(edge) )
    end for
    return MergeComponents( components )
end function
```

```
function MergeComponents(components)
    $pq \leftarrow PriorityQueue()$
    for $component \in components$ do
        $pq.push(component, component.size())$
    end for
    while $pq.size() > 1$ do
        $smallest\_component \leftarrow pq.pop()$
        $mergeable\_component \leftarrow$ find first mergeable component in pq
        $new\_component \leftarrow smallest\_component.merge\_with(mergeable\_component)$
        $components.remove(smallest\_component)$
        $components.remove(mergeable\_component)$
        $pq.remove(mergeable\_component)$
        $components.append(new\_component)$
        $pq.push(new\_component, new\_component.size())$
    end while
    return $components$
end function
```

Figure 11:
Breaking a graph into several loop-free components.

number β_1 mentioned in section 2.

Figure 11 shows our algorithm to break a graph. The idea behind this algorithm is to take every edges as an initial component, then iterate each component to find and merge with a mergeable component until there is no two mergeable components.

E *Determine Paths Homotopy Class*

A continuous function, or a map, $f : X \to Y$ is called a *homotopy equivalence* if there is a map $g : X \to Y$ such that there exist a continuous function $H : X \times [0, 1] \to Y$ with $H(x, 0) = f$ and $H(x, 1) = g$.

Besides using those components to plan motions for a large *C-space*, they

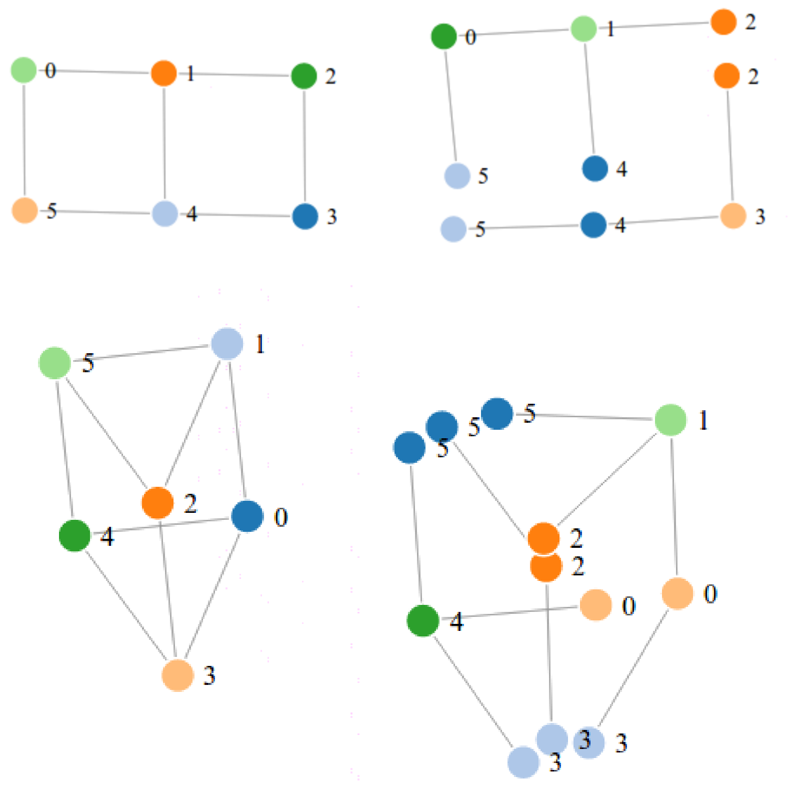

Figure 12: Some examples of breaking a graph. Graphs are visualized using $d3.js$ ($http : //www.d3js.org$)

can also be used to determine if two paths are in the same homotopy class.

Let σ_1 and σ_2 be two paths, with the same start and goal configuration, going through from components g_1 to g_2 via two nodes A and B. σ_1 and σ_2 are homotopy equivalent if $A = B$.

Proof can be found in section 4.

4 Good Properties

Theorem 4.1 *Let S_{init} be the set of initial random configurations. With minimum radius of r_{min}, configurations with clearance larger than r_{min} will be covered, when $|S_{init}| \to \infty$.*

Proof: When the number of initial random configurations approaches infinite, the boundaries of all existing balls are fully covered. So, every configuration not inside any existing balls will be samples as a ball, unless it's less than r_{min} clearance.

Theorem 4.2 *Let B be the set of ball samples got in phase one. Then $|B|$ is finite.*

Proof: We restrict balls to have no less than r_{min} radius and new balls to be centered outside existing ones. Assume we partition the space into $r_{min} \cdot r_{min}$ cells in 2D. The number of cells is finite, if the space is finite. Putting balls with exactly r_{min} radius in the centers of these cells will cover the space totally. The number of balls is finite. Phase one will generate balls no more than the number of balls generated in this way. Figure 9 shows the dual shape of balls generated by previous phase.

Theorem 4.3 *Let $b \in B$ be an element of B, A and B are two configurations in* C-space*. The optimal path σ between A and B is in $\bigcup_{b \in B}$ if σ has clearance no less than $r_m in$.*

Proof This is a direct conclusion of previous theorem.

Theorem 4.4 *Any path σ can be continuously deformed into medial axis.*

Proof This is the definition of Medial Axis $MA(C_{free})$ being a strong deformation retraction of C_{free}.

Theorem 4.5 *Given path σ_1 and σ_2, we say they have the same homotopy type if their image in the topology roadmap is the same.*

Proof Let x be a point in the space X, $B = \{b_1, b_2, ..., b_n\}$ be the set of balls centered on medial axis. Define map $H(x)$:

$H(x) = b_i \quad , if \quad |x - b| < |x - b'|, b' \in B$

This continuous function maps any point in the space to balls centered on medial axis. Dual shape maps any points in the balls to the topology roadmap we build.

Therefore, if two paths have the same image in the topology roadmap. They have the same homotopy type.

Theorem 4.6 *Let g_1 and g_2 be two loop-free graphs such that merging their common nodes will produce a new graph with loops, σ_1 and σ_2, with the same start and goal nodes, be two paths going from g_1 to g_2 via common nodes A and B. σ_1 and σ_2 have different homotopy type if $A \neq B$*

Proof Because A and B are two common nodes shared by g_1 and g_2, if $A \neq B$, merging g_1 and g_2 will create a graph with a loop going from $A \rightarrow B \rightarrow A$. Since σ_1 and σ_2 have the same start and goal nodes, merging them will also create a loop. So σ_1 and σ_2 have different homotopy type.

5 Conclusions and Future Work

The results demonstrated with our algorithm are sound for covering the *C-space* and represent the topology.

However, we don't know what kind of issues there will be when implementing for high dimensional space. The time cost for constructing weighted α-shape(dual shape) of unions of balls is a concern. Because we need to check each *k-simplex* while $k = 1, 2, 3, ..., d$, d being the dimensions. So, our next goal is to implement the algorithm for 3D case, and then higher dimensions.

Our next direction is to study the properties and applications of weighted α-shape, especially applications in bioinformatics and chemistry, then extend to high dimensions.

Another direction will be studying the topology using the topology roadmap we constructed. Its value is underestimated in this paper, and I believe we can find more there.

References

[1] J.-M. Lien, O. B. Bayazit, R.-T. Sowell, S. Rodriguez, and N. M. Amato. Shepherding behaviors. In Proc. IEEE Int. Conf. Robot.Autom. (ICRA), pages 41594164, April 2004.

[2] O. B. Bayazit, G. Song, and N. M. Amato. Enhancing randomized motion planners: Exploring with haptic hints. In Proc. IEEE Int. Conf. Robot. Autom. (ICRA), pages 529536, 2000.

[3] A. P. Singh, J.-C. Latombe, and D. L. Brutlag. A motion planning approach to flexible ligand binding. In Int. Conf. on Intelligent Systems for Molecular Biology (ISMB), pages 252261, 1999.

[4] Delanoue, N.; Jaulin, L.; Cottenceau, B. (2006). "Counting the Number of Connected Components of a Set and Its Application to Robotics". Applied Parallel Computing, Lecture Notes in Computer Science 3732 (1).

[5] Jaulin, L. (2001). "Path planning using intervals and graphs". Reliable Computing 7 (1).

[6] Steven M. LaValle, "Planning Algorithms", Cambridge University Press, ISBN 0-521-86205-1, 2006.

[7] Yeh, Hsin-Yi Cindy, et al. "UMAPRM: Uniformly Sampling the Medial Axis."

[8] LaValle, Steven M. "Rapidly-Exploring Random Trees A New Tool for Path Planning." (1998).

[9] L. E. Kavraki, P. Svestka, L. E. K. P. Vestka, J. claude Latombe, and M. H. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Trans. Robot. Autom., vol. 12, pp. 566580, 1996.

[10] Wilmarth, Steven A., Nancy M. Amato, and Peter F. Stiller. "MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space." Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on. Vol. 2. IEEE, 1999.

[11] S. A. Wilmarth, N. M. Amato, and P. F. Stiller. MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free

space. Technical Report TR98-022, Department of Computer Science, Texas A&M University, College Station, TX, Nov. 1998.

[12] Edelsbrunner, Herbert. ”The union of balls and its dual shape.” Proceedings of the ninth annual symposium on Computational geometry. ACM, 1993.

[13] ”Alpha Shape Introduction”, Duck University, ($http : //biogeometry.duke.edu/software/alphashapes/$)

[14] Edelsbrunner, Herbert; Kirkpatrick, David G.; Seidel, Raimund (1983), ”On the shape of a set of points in the plane”, IEEE Transactions on Information Theory 29 (4): 551559, doi:10.1109/TIT.1983.1056714.

[15] ”Radical Center”, From Wolfram MathWorld – the web’s most extensive mathematics resource. ($http : //mathworld.wolfram.com/RadicalCenter.html$)

[16] Z. McCarthy. Bretl, and S. Hutchinson, ”Proving path non-existence using sampling and alpha shapes,” in IEEE International Conference on Robotics and Automation (ICRA), May 2012.

[17] ”Betti Number”, From Wikipedia – The Free Encyclopedia. ($http : //en.wikipedia.org/wiki/Betti_number$)

[18] Edelsbrunner, Herbert, and John Harer. ”Computational topology: an introduction.” American Mathematical Soc., 2010. page. 88 - 93.

[19] Robins, Vanessa. ”Computational topology for point data: Betti numbers of -shapes.” Morphology of Condensed Matter. Springer Berlin Heidelberg, 2002. 261-274.