Path Clustering with Homology Area

J. Frederico Carvalho\(^1\), Mikael Vejdemo-Johansson\(^2\), Danica Kragic\(^1\) and Florian T. Pokorny\(^1\)

Abstract—Path classification has found many applications in recent years. Common approaches to this problem use aggregates of the distances between points to provide a measure of dissimilarity between paths which do not satisfy the triangle inequality. Therefore, such distance measures need to be used with care to avoid cluster chaining. Furthermore, they do not take into account the topology of the space where the paths are embedded. To tackle this, we extend previous work in path clustering with homology, by employing homology area as a measure of distance between homologous paths in a triangulated mesh. Further, we show that the resulting distance satisfies the triangle inequality, and how we can exploit the properties of homology to reduce the amount of pairwise distance calculations necessary to cluster a set of paths by an order of magnitude.

I. INTRODUCTION

Due to the prevalence of devices with access to positions sensors such as GPS, large datasets of paths have become increasingly available in the past few years [1]. Through path clustering, this data can be partitioned into sets of similar paths, from which motion primitives can be learned in order to control a robot without explicitly programming it [2], [3]. In this paper we present a new method of computing a topologically inspired similarity measure for paths which can be applied to path clustering.

Clustering is a machine learning technique used to classify large sets of data into smaller subsets that are similar to each other [4]. However, the domain of “popular” clustering techniques, such as \(k\)-means and support vector machines, is that of data points in a fixed dimension vector space. Because paths have variable length, they cannot be easily represented in finite dimensions. Therefore, commonly used path clustering methods rely on calculating distances between paths and applying clustering methods that use distance alone [5].

These methods have been successfully applied to different problems, such as learning dynamic scene models, and car trajectories [6]–[8] to name a few. Comparative surveys of commonly used path distances can be consulted in [9], [10]. The aforementioned distances however, do not take into account the topology of the space the paths are drawn from, and therefore can not properly gauge the influence of obstacles on the clustering produced [11]. To tackle this, in [11] we proposed a method for path clustering that only takes the obstacles into account.

In this paper we propose an extension of the framework of clustering with relative homology we proposed in [11].

\(^1\) CAS/RPL, KTH, Royal Institute of Technology, Stockholm, Sweden. \{jfpbdc, dani, ftpokorny\}@kth.se
\(^2\) Mathematics Department, CUNY College of Staten Island, New York, USA. Mikael.VejdemoJohansson@csi.cuny.edu

1 Dynamic Time Warping (DTW) is a popular path distance which can be efficiently computed [13].
2 If the distribution of paths is multimodal, then taking an average of the motions may yield something that lies outside of the scope of the data.
Path clustering has been extensively researched since the early 2000s and has found applications in surveillance, traffic analysis, among others [6]–[9], [14]. In order to cope with the high dimensionality of the space of paths on a vector space, the employed methods use distance-based clustering rather than other methods that require a finite dimensional representation, such as \( k \)-means and support vector machines. An exhaustive review of path clustering combining different distances and clustering methods can be consulted in [15]. As mentioned before, the employed clustering methods rely only on distance between paths, to assign them to a cluster, therefore they need to compare each path to every other. This implies that the path distance needs to be evaluated \( O(N^2) \) times where \( N \) is the number of paths. This fact puts a strain on path clustering methods, since path distances need to be evaluated fast in order to cope with large databases of paths.

Recently in [11] we proposed a clustering method for paths based solely on their homology class. Intuitively, two paths belong to the same cluster (in which case they are said to be “homologous”) if there are no obstacles in the area between them. The strength of this method, lies not only on the speed with which it can be evaluated, but also on the fact that it only needs to do a single pass through the data, making it significantly outperform other methods. However, since this method only distinguishes paths based on topological features, the paths in each cluster present a wide range of variability, we therefore want to be able to cluster paths into more narrow categories in a manner that is compatible with the homology clusters. To this end we use the results from [12] where we presented an efficient method for computing the area between two closed paths as the region whose boundary is the union of the two paths. This notion of distance has the convenience of being compatible with the purely topological clustering in [11].

In this paper we build upon the results in [11], [12] by extending the notion of minimum area homology to quotient complexes, and employing the area of the obtained region as a path distance. Further, we note that in clustering a set of \( N \) paths with \( k \) homology classes, the number of regions that need to be computed can be brought down from \( O(N^2) \) to \( O(kN) \) in the worst case. We show in the experiments section how this allows us to outperform DTW [13] when clustering even relatively small datasets (circa 2000 paths).

### B. Mathematical Background

We now provide a brief description of the mathematical tools used in the remainder of the paper (an exhaustive treatment can be found in [16]). To do this we begin by introducing Homology, which is a general algebraic tool that can be applied to understanding topological spaces, particularly through Simplicial Homology which we explain further on. Finally, we introduce relative homology and homology area which will use to define our path distance.

1) Homology: A chain complex is a pair \((C_\bullet, \partial_\bullet)\), where \( C_\bullet = \{ C_i \}_{i \in \mathbb{N}} \) is a sequence of vector spaces, and \( \partial_\bullet = \{ \partial_i : C_i \rightarrow C_{i-1} \}_{i \in \mathbb{N}} \) is a sequence of linear maps (matrices) called boundary maps (matrices) often depicted in a sequence as in (1).

\[
\cdots \leftarrow C_i \xleftarrow{\partial_i} C_i \xleftarrow{\partial_{i+1}} C_{i+1} \leftarrow \cdots
\]

Where the boundary matrices are required to satisfy \( \partial_i \partial_{i+1} = 0 \). We call an element of \( C_i \) an \( i \)-chain; and when no confusion arises, we denote all boundary maps as \( \partial \), dropping the index.

Given a chain complex, we define two further sequences of vector spaces \( Z_\bullet, B_\bullet \) by:

\[
Z_i = \ker(\partial_i) \quad B_i = \im(\partial_{i+1})
\]

We call elements of \( Z_i \) and \( B_i \) \( i \)-cycles and \( i \)-boundaries, respectively. Further, since \( \partial_i \partial_{i+1} = 0 \) implies \( B_i \subseteq Z_i \), we can define the quotient \( H_i = Z_i/B_i \), which we call the \( i \)-th homology of the chain complex.

Given an \( i \)-boundary \( p \) and an \((i+1)\)-chain \( c \) such that \( p = \partial c \) we call \( c \) the bounding chain of \( p \).

2) Simplicial Homology: Given \( k+1 \) affinely independent points in a vector space \( \{x_0, \ldots, x_k\} \subseteq \mathbb{R}^n \) with \( n > k \), we form the \( k \)-simplex \( \{x_0, \ldots, x_k\} \) (where \( x_0, \ldots, x_k \) is a shorthand for \( \{x_0, \ldots, x_k\} \)). Each such simplex has an orientation that is inherent to the ordering of its points or vertices. Two simplices with the same vertices are said to have opposite orientations if the ordering differs by an odd permutation.

Geometrically, we represent a simplex by the convex hull of its vertices, and its boundary by

\[
\bd([x_0, \ldots, x_k]) = \bigcup_{i=0}^{k} [x_0, \ldots, \hat{x}_i, \ldots, x_k]
\]

Given a topological space \( X \subseteq \mathbb{R}^n \), we can obtain a discrete approximation of \( X \) by taking a discrete set \( \tilde{X} \subseteq X \) and using a triangulation \( T \) of \( \tilde{X} \) (in the remaining we let \( T \) be a subset of the Delaunay triangulation). From \( T \), we can build a chain complex as follows:

In order to construct a chain complex out of \( \tilde{X} \) and \( T \), we denote the ordered set of \( i \)-simplices in \( T \) by \( T^{(i)} = \{ \sigma_{j_0} \} \), and define \( C_i = \mathbb{R}^{b_i} \). Furthermore, for each \( j = 1, \ldots, b_i \) we set

\[
\partial_i c_j = \sum_{k=0}^{i} \pm e_{i,k}
\]

where \( e_{i,b_i} \) and \( e_{i,b_i} \) form the canonical basis of \( C_i \) and \( C_{i-1} \), respectively; and \( e_{i,k} \) is the index of the \( k \)-th face of \( \sigma_j \) and the sign depends on the orientation.

Thus, we obtain a simplicial complex (see [16] for a proof), which we denote by \((C_\bullet(X), \partial_\bullet)\) and its homology by \( H_\bullet(X) \).

In this setting, it can be observed that \( 1 \)-cycles correspond to sums of edges that form a closed path, and that \( 1 \)-boundaries correspond to those \( 1 \)-cycles that form the boundary of a closed region (see Figure 2 for an example).

3) Relative Homology: Given a chain complex \((C_\bullet, \partial_\bullet)\) we define a subcomplex as a complex \((A_\bullet, \partial_\bullet)\) where for each \( i, A_i \) is a subspace of \( C_i \) and \( \partial_i \) is the boundary map of \((C_\bullet, \partial_\bullet)\) restricted to \( A_i \). From such a subcomplex we can
II. Methodology

We start this section by presenting and describing the algorithms that make up the main contributions of this paper as well as provide the complexity of Algorithm 2. In the remainder of the section we introduce the theory that ensures the correctness of our algorithms.

A. Algorithms

We now present the algorithm to compute the quotient of a simplicial complex (Algorithm 1), and to compute the homology-area of associated to a set of paths (Algorithm 2).

Algorithm 1: Computing a quotient where the support of a characteristic function is collapsed to a single point

\begin{verbatim}
input : S /* simplicial complex */
input : q /* characteristic function */
/* q satisfies q(x) = 1 for x \in A, */
/* and q(x) = 0 otherwise. */
/* S = (points, tris) where: */
/* pts: list of 2D points. */
/* tris: list of triangles, each is */
/* a triple of indices in pts. */
output: Q /* simplicial complex */
/* Q is a representative of S/A */
1 qPts = (Null);
   /* 1st is the image of A */
2 for p \in pts do
3   if q(p) = 0 then
4     append p to qPts;
5   qTris = {};
6   for tri \in tris do
7     if q(pts[tri]) = 1 then
8        has0 = True;
9     else
10        /* include 0 at most once */
11        append qv to qTri;
12        if has0 then
13           /* prepend 0 to qTri; */
14           prepend 0 to qTri;
15    return Q = (qPts, qTris)
\end{verbatim}

In Algorithm 1 we assume the simplicial complex S is given as a pair \((\text{points}, \text{tris})\) where \(\text{points} \subseteq \mathbb{R}^2\) is a (finite) set containing the vertices of the simplicial complex, and \(\text{tris}\) is a triangulation of those points, i.e. \(\text{tris} = S^{(2)}\). Similarly, the quotient Q is represented by a pair of points and triangles \((q\text{Pts}, q\text{Tris})\).

The subset \(A \subseteq S\) so that \(Q = S/A\) is passed to Algorithm 1 as a characteristic function \(q : \mathbb{R}^2 \rightarrow \{0,1\}\) such that \(q(x) = 1\) if and only if \(x \in A\).
The algorithm proceeds by first letting $q_{Pts}$ have Null as its first point, as this will be the image of every point in $A$ and therefore has no (unique) correspondence with a point in $S$, and then adding every point that is not in $A$. Secondly in the lines 6–16 it translates the triangles in $q_{Tri}$ to triangles with the corresponding vertices in $q_{Pts}$. This translation is done in lines 9–14, where the indices in a triangle $tri$ are replaced in $q_{Tri}$ by their counterparts from $q_{Pts}$ unless the point is in $A$; subsequently if $tri$ contains a point in $A$ the index 0 is added to $q_{Tri}$ in line 16, so that it is added at most once. The resulting simplex $q_{Tri}$ is then added to $q_{Tris}$, and $Q = (q_{Pts}, q_{Tris})$ is returned.

Now we proceed to show how we can use the quotient complex and bounding chains to efficiently calculate a distance array using homology area in Algorithm 2. As we note in Section III even though the process of calculating a bounding chain is slow, calculating the distance array can be done efficiently due to Corollary 1.

Algorithm 2 comprises two phases, phase one in lines 1–16 performs the heavy calculations (i.e. it computes the necessary bounding chains). Phase two, in lines 17–27 uses these bounding chains to compute the array of distances between paths.

This algorithm uses the subroutine $BoundingChain(p,q)$ which, given two 1–chains $p, q$ calculates the bounding chain of $p − q$ in case they are homologous, and reports a failure otherwise. In our implementation, we use a least squares algorithm to solve equation (2) (which we justify in Section II-B), and intuitively, given the innate sparsity of the boundary matrix, an alternative would be to use a simple row-reduction algorithm, but the associated change of base matrix will be dense, making it more computationally expensive.

In phase one the algorithm builds up three lists $PathCluster$, a list that at position $i$ it has the homology class of path $Paths[i]$; $HomologyClusters$, a list of lists where the elements of $HomologyClusters[i]$ are the indices of paths in the $i$–th homology class. The third list, $BoundingChains$, stores, at position $i$ the results of $BoundingChain(p,q)$ where $p$ is the first path in the $i$–th homology cluster, and $q$ ranges over the paths homologous to $p$, in order of appearance in $Paths$.

In phase two, the algorithm uses lists built up in phase one to produce the array of distances $Distarray$. To achieve this it checks what is the homology cluster of path $i$ (line 19) and computes the distance to path $j$ for every $j > i$ by checking if they have the same homology class; if they are, it uses Corollary 1 (which is summarized in Figure 4) to calculate the distance (lines 21–25), otherwise, it stores the value 0 (line 27).

Algorithm 2: Distance array using homology area.

```plaintext
Algorithm 2: Distance array using homology area.

input : Paths /* a list of paths */
output: An array of pairwise distances
1 PathCluster = {};
2 BoundingChains = {};
3 HomologyClusters = {};
4 for $i \in \{0, \ldots, \#Paths\}$ do
  5    for $Cluster \in HomologyClusters$ do
    6        /* check if the current path is in any existing cluster */
    7            $p = Paths[i]$;
    8            $q = Cluster[0]$;
    9            $j = Index(Cluster, HomologyClusters)$;
    10           $Chain = BoundingChain(q, p)$;
    11           if successful then
    12                    append $j$ to PathCluster;
    13                    append $i$ to Cluster;
    14                    append $Chain$ to BoundingChains[$j$];
    15           if no cluster was found then
    16                /* not homologous to any already seen path, new cluster */
    17                append $\{i\}$ to HomologyClusters;
    18                append $\{\}$ to BoundingChains;
  19    /* use the bounding chains to compute pairwise distances */
  20  $Distarray = \{\}$;
  21  for $i \in \{0, \ldots, \#Paths\}$ do
  22      $c = PathCluster[i]$;
  23      for $j \in \{i+1, \ldots, \#Paths\}$ do
  24          if $PathCluster[j] = c$ then
  25              $k = index$ of $i$ in $HomologyCluster[c]$;
  26              $l = index$ of $j$ in $HomologyCluster[c]$;
  27              $Chain = BoundingChains[c, k - 1] = BoundingChains[c, l - 1]$;
  28              append area($Chain$) to $Distarray$;
  29          else
  30              append $\infty$ to $Distarray$;
  31  return $Distarray$;
```

Now, note that finding the minimum area bounding chain amounts to solving a $k \times l$ linear system, where $k, l$ are respectively the number of edges and faces of the simplicial complex. If we let $\mu = \max(k, l)$, such a system can be solved by a linear solver in $O(\mu^2)$. From analyzing algorithm 2 we see that when analyzing each path, we need to compute at most $m$ bounding chains, where $m$ is the number of elements of $HomologyClusters$ (line 5). Therefore, the
computing the first relative homology group $H_1(X, A)$ we define instead the homology class of 1–chains with boundary contained in $A$, which correspond to paths beginning and ending in $A$.

\textbf{Remark 1.} By Theorem 1 we can compute the homology group $H_1(X/A)$ instead of $H_1(X, A)$ as these are canonically isomorphic. This has a practical advantage, since $X/A$ has less simplices then $X$, which reduces the dimension of the boundary matrices $C_1(X/A), C_2(X/A)$ relative to those for $C_1(X, A)$ and $C_2(X, A)$, making the computation of solutions to (3) more efficient.

Algorithm 1 shows how one can create a quotient complex from a simplicial complex and the characteristic function of a closed region $A$.

Now, given a 2–dimensional simplicial complex $X$ and a subcomplex $A$ we note that, any 2–simplex in $\sigma \in (X/A)^2$ is a quotient of some 2–simplex $\sigma' \in X^2$ and hence we can define $\text{Area}(\sigma) = \text{Area}(\sigma')$, which then allows us to implement minimum area homology on $X/A$.

In [12] we presented a framework to compute a minimum area bounding chain on a general simplicial complex by computing the solution to equation (2), and propose several methods by which this may be achieved. Particularly, computing least squares solutions as this can be done efficiently on sparse systems. However, often fast methods, such as least squares, may yield chains that do not have minimal area. In what follows, we prove that if the chain complex $X$ can be embedded in $\mathbb{R}^2$, then for any 1–cycle there is at most one bounding chain, thereby proving that in such cases approximate methods yield exact solutions. In Proposition 2 we generalize to quotients over sets $A$ with no holes.

\textbf{Lemma 2.} Any simplicial complex $X$ which can be embedded into $\mathbb{R}^2$ satisfies $H_2(X) = 0$.

\textbf{Proof:} Let $X$ be a simplicial complex that can be embedded into $\mathbb{R}^2$ and $X^c$ denote (a triangulation of the closure of) its complement, then using the Mayer-Vietoris sequence of $\mathbb{R}^2 = X \cup X^c$, we have:

$$\cdots \to H_3(\mathbb{R}^2) \to H_2(X \cap X^c) \to H_2(X) \oplus H_2(X^c) \to H_2(\mathbb{R}^2) \to \cdots$$

Now, recall that $H_i(\mathbb{R}^k) = 0$ for $i \neq 0$ ([16]). Note also that $X \cap X^c = \partial X$ is a 1–dimensional subcomplex of $X$ and $X^c$, so $H_2(X \cap X^c) = 0$ (as it has no 2–cells), hence we conclude that the sequence:

$$0 \to H_2(X) \oplus H_2(X^c) \to 0$$

is exact, and $H_2(X)$ is a summand of 0 and hence $H_2(X) = 0$.

\textbf{Proposition 2.} Let $X$ be a simplicial complex which can be embedded in $\mathbb{R}^2$, and $A$ a subcomplex of $X$ such that $H_1(A) = 0$. Then, given any 1–cycle $c$, there exists, at most one 2–chain $s$, such that $\partial s = c$.  

\[\text{Definition 1.} \text{ Define the homology area path distance on } X \text{ as} \]

\[d_{H(X)}(p, q) = \min_{c = p \to q} \text{area}(c) \]  

where $\min_{x \in \mathbb{R}} x = +\infty$.  

\textbf{Proposition 1.} Let $X$ be a simplicial complex where each 2–simplex has non-negative area, then $d_{H(X)}$ is symmetric and satisfies the triangle inequality, i.e.:  

\begin{itemize}
  \item $d_{H(X)}(p, q) = d_{H(X)}(q, p)$
  \item $d_{H(X)}(p, q, r) \leq d_{H(X)}(p, r)$
\end{itemize}

\textbf{Proof:} Let $c$ be a bounding chain of $p - q$, then due to linearity of $\partial$, $\partial(-c) = q - p$ and $\text{area}(c) = \text{area}(-c)$ meaning that $d_{H(X)}(p, q) = d_{H(X)}(q, p)$.  

Assume $p, q$ and $r$ are homologous pairs of paths, then there exist 2–chains $c_{p,q}$ and $c_{q,r}$ such that $\partial c_{p,q} = p - q$ and $\partial c_{q,r} = q - r$. This implies that $\partial(c_{p,q} + c_{q,r}) = p - q + q - r = p - r$ and therefore $c = (c_{p,q} + c_{q,r})$ is a bounding chain for $p - r$. Furthermore we see that $c_i \neq 0$ only if $(c_{p, q})_i \neq 0$ or $(c_{q, r})_i \neq 0$ and therefore area$(c) \leq$ area$(c_{p, q}) +$ area$(c_{q, r})$, which implies $d_{H(X)}(p, r) \leq d_{H(X)}(p, q) + d_{H(X)}(q, r)$.

This defines a distance between edge paths that is only finite for paths that are homologous. A shortcoming of this, is that only homologous paths that have the same first and last points may have finite area. However, this can be mitigated using relative homology.

More specifically, we recall from [11], that by defining goal regions $A$ in the space $X$ (associated to a subcomplex) and

\text{remark:} that is, if the $i$–th edge is of the form $[e, e']$ then $\tilde{p}_i = p_{+}_i - p_{-}_i$ where $p_{+}_i$ is the number of times the edge is traversed from $e$ to $e'$, and $p_{-}_i$ is the number of times the edge is traversed in the from $e'$ to $e$.
Proof: To begin, consider the long exact sequence in homology of the pair \((A,X)\):

\[
\cdots \rightarrow H_{i+1}(A) \rightarrow H_{i+1}(X) \rightarrow H_{i+1}(X,A) \rightarrow H_i(A) \rightarrow \cdots
\]

Particularly for \(i = 1\), \(H_i(A) = 0\) and so we have:

\[
\cdots \rightarrow H_2(A) \rightarrow H_2(X) \rightarrow H_2(X,A) \rightarrow 0 \rightarrow \cdots
\]

So, using Lemma 2 we know that 
\(H_2(A) = H_2(X) = 0\) and so \(H_2(X,A) = 0\). Finally, given any 1-cycle \(c\) let \(s,s'\) be two 2–chains satisfying \(\partial s = \partial s' = c\), then \(t = s - s'\) is a 2–cycle, and since \(H_2(X,A) = 0\), we conclude that \(t\) is the boundary of some 3–chain. However, since \(X\) can be embedded in \(\mathbb{R}^2\), it has no 3–simplices, from which we can conclude that \(t = 0\) and \(s = s'\).

We now show how we can use this result to quickly calculate bounding chains between paths.

Corollary 1. Let \(X\) be a two-dimensional simplicial complex and \(A\) a subcomplex with \(H_1(A) = 0\), then for three one
chains \(p,q,r\) such that \(c_{p,q}\) and \(c_{p,r}\) are the minimum area bounding chains of \(p-q\) and \(p-r\), respectively, then \((c_{p,r} - c_{p,q})\) is the minimum area bounding chain of \(q-r\).

Proof: To prove the corollary from the proof or Proposition 1 that \((c_{p,r} - c_{p,q})\) is a bounding chain of \(q-r\), (this comes from linearity of the boundary operator). Furthermore, from Proposition 2 if there is a bounding chain of \(q-r\) then it is unique, therefore, \((c_{p,r} - c_{p,q})\) must be the minimum area bounding chain of \(q-r\).

Finally, since our main goal is to cluster paths on a simplicial complex, we note that many algorithms used for path clustering rely on computing distances between all pairs of paths. Specifically, if we want to cluster the paths \(p_0, p_1, \ldots, p_N\) with a distance function \(d\), then we need to compute:

\[
\text{Distarray} = [d(p_0, p_1), \ldots, d(p_0, p_N), d(p_1, p_2), \ldots, d(p_1, p_N), \ldots, d(p_{N-1}, p_N)]
\]

Which corresponds to the entries of the matrix \([d(p_j, p_j)])_{i,j=0}^N\) that lie above the main diagonal. In the case that the distance function is hard to compute, as it often is in the case of path distances ([18]), producing such a distance array can take a long time. Corollary 1 provides a way to mitigate this cost when using the homology area distance function, and we exploit this fact in Algorithm 2 to avoid calculating unnecessary bounding chains.

### III. EXPERIMENTS

To test our framework we used two distinct environments. First we created a synthetic dataset described in Section III-A and tested our framework on it. Secondly, in Section III-B we used our framework on real data, namely on the set of tracks of pedestrians on the computer science forum at Edinburgh University [19], a subset of which can be seen in Figure 6a.

#### A. Toy example

The space created for the synthetic dataset comprises a bounded region of the plane with two holes. The simplicial complex was obtained by random sampling on this area. The relative region we chose comprises two rectangles along the side so that we can observe different clusters of paths going left to right.

In order to be able to observe the characteristics of clustering by relative homology previously identified in [11], as well as the distinction between different sub-clusters (using minimum area homology), we synthesized paths satisfying:

- Each path begins in one of two small areas in the relative region to the left (we call them \(O_1\) and \(O_2\)).
- Each path ends in one of two small areas in the relative region to the right (we call them \(T_2\) and \(T_3\)).
- Each path passes through a preselected (randomly chosen) point in a neighborhood of the middle vertical axis of the space (comprising three connected components which we call \(M_1, M_2\), and \(M_3\)).

Paths were then computed by randomly drawing a triple from \(O_i \times M_j \times T_k\), and interpolating these points with a Gaussian process. By choosing the paths in this way we have access to a ground truth model which is simply given by the triple \((i,j,k)\).

As expected, we observed that relative homology alone was enough to detect the different clusters depending on \(j\), furthermore, using homology area we were able to fully recover the ground truth, as seen in Figure 5. Homology area allows us to distinguish between the four clusters with \(j = 2\) which cannot be separated with homology alone. Furthermore, for comparison we used FastDTW [13] from which we obtained similar results.

#### B. Edinburgh dataset

In this experiment we randomly selected 2000 paths from the dataset of trajectories collected in the computer science forum of Edinburgh University (as seen in Figure 6a). From these paths we constructed a simplicial complex that fully covered the area around them as shown in Figure 6b, and the relative regions were chosen so that they contain the endpoints of most paths (those that were not encompassed...
TABLE I: Time elapsed in computing Distarray with DTW, FastDTW, and Homology area. “Average time” denotes the average amount of time to compute the distance between two paths, whereas “Total time” denotes the total amount of time spent in computing the distance array.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average time (ms)</th>
<th>Total time (h:m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homology Area</td>
<td>90.809</td>
<td>0 h 17 m</td>
</tr>
<tr>
<td>FastDTW</td>
<td>5.301</td>
<td>2 h 42 m</td>
</tr>
<tr>
<td>DTW</td>
<td>18.152</td>
<td>10 h 21 m</td>
</tr>
</tbody>
</table>

In order to test the performance of our algorithm we computed the array of distances using DTW, FastDTW [13] and our implementation of Homology area, and noted the times in Table I. These tests were performed using a single core of an Intel Core i7-4790K @ 4.00GHz with 32Gb RAM. Our algorithms were fully implemented in Python, and for comparison we used the implementations of DTW and FastDTW from [20]. From these results we observe that Homology area outperforms (in Total time) DTW and FastDTW. The reason for this is that even though it takes more time to calculate each individual pairwise distances with Homology Area, by using Algorithm 2 most of the bounding chains needed to compute the distance array are obtained as linear combination of other bounding chains.

In this paper we were able to show that we can extend topological methods for path clustering to provide a compatible path distance. Furthermore, we were able to observe that the proposed method is able to cope with large numbers of paths by leveraging the fact that not every distance needs to be computed from scratch, thereby reducing the time needed to produce a full distance array (essential for distance-based clustering) by an order of magnitude.

Acknowledgements: The authors would like to thank Erin Wolf Chambers for her valuable contribution to the research and structure of this paper.

REFERENCES