Available online at www.sciencedirect.com

ScienceDirect

IFAC-PapersOnLine 49-15 (2016) 224–229

Matching Paths in Topological Maps

Sören Schwertfeger * **Tianyan Yu** **

* *ShanghaiTech University, No. 8 Building, 319 Yueyang Road,*
Shanghai 200031, China (e-mail: soerensch@shanghaitech.edu.cn)
** *ShanghaiTech University, (e-mail: yuty@shanghaitech.edu.cn).*

Abstract: Topological maps have many applications in robotics. Matching two topological maps from the same environment can be used for map merging, place detection, map evaluation and other purposes. In this paper we present an approach to match two corresponding edges from two Topology Graphs to each other based on the actual path with which the vertices of the edges are connected in the underlying 2D grid maps. We perform experiments with two artificial maps as well as with four maps from the RoboCup Rescue WorldCup 2010.

Keywords: Mobile robots, Mapping, Topological map, Path matching, Voronoi graph

1. INTRODUCTION

In many robotic applications mapping is an essential task for mobile robot systems. The generated maps are models of the environment which are often represented as 2D grid maps. This image like map format is quite detailed. Topology Graphs are more abstract representations which only comprise of places and connections between them.

There are many applications for topological maps, for example for map merging Saeedi et al. (2014), place detection Beeson et al. (2005), or planning Thrun (1998). There are also different ways to generate topological representations from 2D grid maps, for example based on thinning methods Ko et al. (2004) or Voronoi Diagrams Kolling and Carpin (2008), Lau et al. (2010).

All maps have some degree of error which should be measured Schwertfeger et al. (2011). Recent work on map quality assessment matches the topology graph of a ground truth map to the topology graph of robot generated maps Schwertfeger and Birk (2015a), Schwertfeger and Birk (2013). Those algorithms base the matching on a similarity metric of the vertices and a common sub-graph isomorphism. In this work we explore the possibility to match the edges of the Topology Graphs instead of the vertices. The resulting edge matching is useful not only for map evaluation but also to other applications like map merging or place recognition.

This paper is structured as follows: Section 2 provides a short review of related work while Section 3 introduces definitions and the algorithm. The experimental validation is presented in Section 4 and is followed by the conclusions.

2. RELATED WORK

Previous work on path or skeleton matching mainly follows three different approaches:

- Map matching with shortest geodesic paths
- Contour partitioning and skeleton pruning
- Shape matching using Bayesian formula with skeleton similarity

In Bai and Latecki (2008), only the endpoints of a path are considered. As the endpoints are always on the contour, they are indexed in a clockwise orientation. The least distances through the skeleton path for each endpoint pair, which is called "shortest geodesic path", are calculated, and vertices are sampled from the path equidistantly. A normalized vector is used to save the radii of each sampled vertex's maximal disc. With this vector, the similarity between two paths can be calculated. For skeleton matching, the similarity matrix between a certain vertex v_i from map A and another vertex v'_j from map B is calculated by going through all shortest geodesic paths which are connected to the vertex v_i from map A and calculating the similarity with all shortest geodesic paths which are connected to the vertex v'_j from map B.

By applying optimal subsequence bijection (OSB), some outlier endpoints are filtered out, and the dissimilarity between two endpoints are obtained. A dissimilarity matrix is then generated by going through all endpoints in both map A and map B. By using the Hungarian algorithm, each endpoint from map A finds its best match in map B. It has to be noted that in this approach the matching of connection vertices which are connected to 2 edges and the junction vertices which are connected to 3 or more edges are not taken into account because the endpoint which is connected to only 1 edge is always on the contour, which saves a lot of information about the shape of the object and this is very important for matching.

If applied to the figures like horses or humans, whose skeletons have many branches on the contour, and each vertex on the algorithm finds the shortest geodesic paths to the other on the contour, the approach works well. For robot maps, there may be very few dead-end vertices on the contour, and the paths in the maze-like map are also complicated. So the shortest geodesic path between vertices on the contour seems not suitable for robotic map matching.

In their another approach Bai et al. (2007), skeleton maps are used for contour partition. The Discrete Curve Evolution (DCE) Latecki and Lakämper (2000) is applied to the

10.1016/j.ifacol.2016.07.736

contours of objects in digital images. It uses the contour pixels to form a polygon and each time a contour vertex with the least contribution to the contour is removed. At the same time, skeleton pruning is carried out, in which only the skeleton points whose generating points belong to different contour partitions are reserved. By setting a certain threshold to stop the pruning iteration and removing concave vertex on the contour and the skeleton edge connected to it, several main skeleton edges and the partition of the contour are obtained.

Since only the main part of the skeleton map and its convex hull (in the form of polygon) is preserved, this approach can be further used for shape classification (by matching the pruned skeleton map).

In this approach, the figure always has a contour (such as leaf), contour partitioning and skeleton pruning is used to get the main skeleton of it. In contrast, the skeleton of robotic maps is much more complicated. Some paths are in a room, others connect one room to another. This approach can only be applied to the paths in a certain room in the map, since room has a contour, but it can not applied on the whole map.

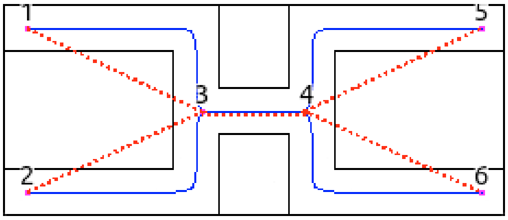
Yang and Sze (2007) use the longest weighted path in a directed acyclic graph to match two graphs, and find the top K suboptimal paths in polynomial time. This approach is used in biology, and matches the abstract graph and paths.

3. ALGORITHMS

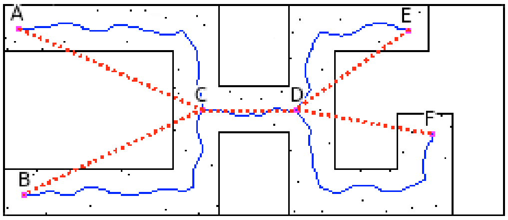
First we define a terminology for the following sections.

- **Graph:** A graph is an ordered pair $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ comprising a set $\mathbf{V}$ of vertices or nodes together with a set $\mathbf{E}$ of edges or links.
- **Topology Graph:** A graph in which vertices represent locations and edges which represent the fact that there exists a drivable route between two vertices.
- **Vertex:** A node in the graph. It is attributed with the metric location as x, y coordinates.
- **Edge:** A (drivable) connection between vertices in the graph. Each edge is attributed with a metric Path.
    - · Half Edge: In a Doubly Connected Edge List, a half edge is a directed connection between two vertices.
    - · Twin: Each half edge has one twin. A half edge's source vertex is its twin's target vertex and vice versa.
- **Path:** Every edge is attributed with exactly one path. The path represents the metric information of a free (drivable) way between two vertices. A path is a small directed graph consisting of a series of edges which are connected in a (possibly curved) line between the source vertex and the target vertex of the parent edge. Each vertex in the path is attributed with its metric x, y coordinates.

The path matching works on the Topology Graphs introduced in detail in Schwertfeger and Birk (2015a) and Schwertfeger and Birk (2013). There, a Voronoi Diagram is used to generate a topological graph from a 2D grid map. After some filtering of vertices and edges, the topological graph can directly be used to match two Topology Graphs with each other. In that work the matching was based on similarity metric of the vertices and a common sub-graph isomorphism. In contrast, this paper explores the matching based on the edges of a topology graph, or more precisely based on the path attributed to each edge.



(a) Map I: Artificial ground truth map.



(b) Map II: Artificial map with man-made noise and difference around E and F.

Fig. 1. Artificial 2D grid maps (black) with Topology Graphs (labeled vertices: purple, edges: red dotted, paths attributed to edges: blue)

Instead of using a Voronoi Diagram other methods of generating the Topology Graph are possible, for example thinning methods Ko et al. (2004). The Topology Graph generation used here works best in relatively confined environments. In open areas graphs often differ between different maps and it is thus difficult to match those graphs. Our work on generating Topology Graphs which represent such open areas as vertices in the graph (”room detection”) and thus alleviate this problem will be published soon. Also it should be noted that the paths used here are not paths that the robot actually traversed. Our edges/ paths are connections between vertices have been found to be possible to drive for a robot - by evaluating the metric grid maps that are the sole input to this algorithm.

Please observe the definition: An edge is a topological graph element directly connecting two vertices (”in a straight line”), while the path is a metric attribute of an edge which forms (an often curved) line in the free space of the environment between the two vertices of the edge. For example in Figure 1(a) there is an edge between vertices 1 and 3 (red dotted line) and a curved path attributed to this edge (blue line). In the other figures the red dotted edges are omitted. Also note that the topology of the graph is undirected, but the graph is represented using directed half edges and each half edge has a corresponding twin in which the source and target are switched.

3.1 Path Matching

In the following algorithms we assess the dissimilarity of two paths and thus also the dissimilarity of their associated edges. The path dissimilarity is calculated by sampling each path equidistantly. The result is a list of (x, y) 2D coordinates. After sampling, the 2D Horn's algorithm