Weighted Euclidean Steiner Trees for Disaster-Aware Network Design

Luis Garrote[1,2], Lúcia Martins[1,3], Urbano J. Nunes[1,2], Martin Zachariasen[4]
[1] Electrical and Computer Engineering Department, University of Coimbra, Coimbra, Portugal
[2] Institute of Systems and Robotics, Coimbra, Portugal
[3] INESC Coimbra, Portugal
[4] IT University of Copenhagen, Denmark
Email:{garrote,urbano}@isr.uc.pt,lucia@deec.uc.pt,marz@itu.dk

***Abstract*—We consider the problem of constructing a Euclidean Steiner tree in a setting where the plane has been divided into polygonal regions, each with an associated weight. Given a set of points (terminals), the task is to construct a shortest interconnection of the points, where the cost of a line segment in a region is the Euclidean distance multiplied by the weight of the region. The problem is a natural generalization of the obstacle-avoiding Euclidean Steiner tree problem, and has obvious applications in network design. We propose an efficient heuristic strategy for the problem, and evaluate its performance on both randomly generated and near-realistic problem instances. The minimum cost Euclidean Steiner tree can be seen as an optical backbone network (a *Spine*) avoiding disaster prone areas, here represented as higher cost regions.**

***Index Terms*—Euclidean Steiner Tree, Heuristic, Communication Networks Reliability**

I. Introduction

The *spine* concept was introduced in [1] as a framework to obtain in an efficient manner, i.e. without over-engineering the network, high availability for critical services (such as smart grid communications). A spine is a subset of links and nodes with high availability, at the physical layer of the network, that together with lower availability network elements can offer redundancy to fulfill the given availability requirements. Although it is a more general concept, the most straightforward approach is to associate the spine to the optical backbone network and, if all pairs of nodes are involved in critical services, to consider a spanning tree as a spine [1]. However, other subgraphs, different from a spanning tree, can be considered as a spine. In [2] the constructed spine also guarantees that in case of a wide-area disaster with a maximum diameter, the network will not disconnect: for each pair of nodes (not affected by the disaster), there exist two geodiverse disjoint paths in the spine — each with the desired availability — such that at least one of these paths is outside of the disaster area. Other approaches to improve network resilience based on geodiverse routing can be found in [3], [4]. A general overview of strategies to improve network resilience, and to avoid the collapse of communication networks in case of large-scale natural disasters, is presented in [5].

In this work the goal is to design a spine from scratch connecting a set of nodes that represent communication equipment, while avoiding disaster prone areas, through a minimum length infrastructure. The classical *Euclidean Steiner tree problem* in the plane is to construct a minimum-length tree that interconnects a given set of points in the plane (terminals). Computing a Euclidean minimum Steiner tree is a well-known NP-hard problem [6]. In order to minimize length, additional nodes, so-called Steiner nodes, may be added to the solution tree. The *Euclidean Steiner tree problem with obstacles* is a generalization of the classical problem where a set of solid obstacles (polygonal regions in the plane) must be avoided by the solution tree. Exact algorithms for the Euclidean Steiner tree problem with and without obstacles are presented in [7], [8].

Geographical areas more prone to natural disasters such as hurricanes, floods or earthquakes can often be associated with a probability of occurrence of such disasters in some amount of time. Such hazard maps can be used to classify into the same shared risk group (SRG) a set of links or nodes if they are vulnerable to the same disaster. The use of SRG-disjoint active and backup paths can be an effective strategy to improve network resilience in case of disasters [9], [10]. In this work, we propose to represent disaster prone areas in the Euclidean plane as *soft* obstacles, i.e. obstacles that can be traversed with a given cost per unity of length, and the purpose is to obtain a low-cost Euclidean Steiner tree avoiding these disaster prone areas.

The Euclidean Steiner problem with soft obstacles is a generalization of the obstacle-avoiding Euclidean Steiner problem, and it has been considered in the context of rectilinear Steiner trees [11]. (The rectilinear Steiner problem is a special case where the tree edges must form right angles.) A variant of the Euclidean problem was considered by Frommer, Golden and Pundoor [12]; in this problem a Euclidean Steiner tree should be constructed on a grid, where each node has an associated cost.

The simpler problem of constructing a shortest path in the presence of soft obstacles was introduced by Mitchell [13]; the first exact algorithm for this problem was given by Mitchell and Papadimitriou [14]. Rowe and Richbourg [15] described and implemented the first practical algorithm for the shortest path problem.

In this paper we propose a heuristic based on a local search strategy to solve the Euclidean Steiner problem with soft obstacles. To the best of our knowledge, this is the first attempt to address this NP-hard problem. The purpose is to obtain a low-cost tree representing the backbone network (the spine) taking into account disaster prone areas, which are represented in the Euclidean plane as polygons. Note that the polygonal representation of the obstacles is not a limitation because is is always possible to accurately approximate other shapes by polygons. Also note that additional Steiner nodes that appear in the solution may in practice have additional cost if communication equipment is required in those nodes. In the context of this work, those nodes will have no additional cost. The obtained tree corresponds to the final layout of fibers or microwave links of the transport network with high availability (guaranteed by the quality and redundancy of the equipment that should be deployed). Note that the final network topology will have additional links which may have lower availability and are not considered in this work.

The paper is organized as follows. In Section II, the proposed heuristic approach is presented; the section is subdivided into three subsections: in subsection II-A the computation of shortest paths between two points in the presence of soft obstacles is described; in subsection II-B a meta-heuristic to solve the 3-points problem is proposed; and in subsection II-C the proposed heuristic to obtain a low-cost Euclidean Steiner tree is presented. In Section III the validation of the proposed heuristic is presented with two distinct scenarios and the obtained results are discussed. Conclusions are presented in Section IV.

II. Steiner tree Approach

The Euclidean Steiner tree problem with soft obstacles (ESMT-SO) is defined as follows: given a division of the plane into polygonal regions with weights assigned to each region and a set of terminals $t \in T : t = 1, \ldots, |T|$ to be interconnected, compute a weighted Euclidean minimum Steiner tree. The terminal nodes are defined by 2D Cartesian points $p_t = (x_t, y_t)$. The cost of the tree is defined as the Euclidean distance multiplied by the underlying weight of each region.

In general, obstacles can be divided into two classes: solid and soft obstacles. Solid obstacles are non-traversable obstacles, i.e. regions that cannot be crossed, whereas soft obstacles can be crossed with a multiplicative length-increase corresponding to its weight. The class of soft obstacles can be further divided into homogeneous and non-homogeneous obstacles. In homogeneous obstacles the cost is constant, and for non-homogeneous obstacles the cost varies accordingly to a predefined model (e.g., a Gaussian function).

The present work focuses on *homogeneous soft obstacles*, leaving the non-homogeneous for future work. A soft obstacle is a polygon defined by its extreme points. A homogeneous soft obstacle $o \in \mathcal{O}$ is defined by a set of 2D Cartesian points $p^o_k = (x^o_k, y^o_k)$, $(k = 1, \ \ldots, \ n^o)$ and $p^o_1 = p^o_{n^o}$, and an associated weight c_o where $o \in \mathcal{O}$. Note that the soft obstacles

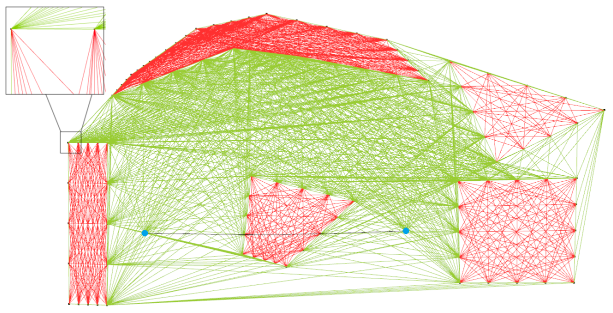


Fig. 1. Representation of a shortest path between two terminal points (blue dots) passing through a soft obstacle in a scenario with five soft obstacles. In red, connections inside the obstacles and in green connections outside the obstacles.The zoomed section shows that the obstacle's boundary corresponds to a connection outside the obstacle.

are given as non-self-intersecting polygons and that obstacles may overlap in their boundaries; in this case the weight is the highest among the overlapping polygons.

A. Shortest Paths

The proposed shortest path approach (**ShortestPath_SoftO**), based on [16], converts the Euclidean shortest path problem with soft obstacles into a graph problem by uniformly discretizing the obstacle's edges. Therefore a set of points, known as portals (see Fig. 1), is placed along the edges of the polygons creating a graph where the shortest path is computed using Dijkstras algorithm.

Given two nodes, the shortest path considering the soft obstacles requires the computation of all the edges between all obstacles and between the two nodes. In Fig. 1 a representation of the two points problem showing the connections over the obstacles, between obstacles and between each terminal and the obstacles is presented. In this work, connections on the boundary of obstacles are computed with the cost of the region outside of the obstacle or the highest cost in the case of overlapping polygons.

The shortest path algorithm in this work employs Dijkstra algorithm (recurring to a heap). The edges correspond to all the connections between obstacles and the two terminal nodes (see Fig. 1). Computing all connections is time consuming and requires heavy computational resources for scenarios with multiple obstacles; therefore, some heuristic approaches are considered to tackle this problem.

In order to avoid the computation of shortest paths between two points that have one or more obstacles in between, meaning that the points are not visible to each other, it is necessary to compute the intersection between the two points and the edges of the obstacles. This computation is heavy, particularly in the case where the polygons are non convex. To speed up the computation we propose an intermediary representation for each obstacle in the form of a 2D grid (see zoomed area in Fig. 2a). A 2D grid is defined as a two-dimensional ($2D$) environment representation bounded between ($[0, \ n_{rows}], [0, \ n_{cols}]$), composed by a set of cells c_{ij} with row i and column j. Each cell c is defined with constant dimensions s_c^o but is adjusted dynamically depending on the size of the polygon and the minimum granularity of the map. The anchor of the grid is defined in the geometric centroid of the polygon $p_c^o = (x_c^o, y_c^o)$. Each cell in the grid contains a value corresponding to one of the following labels: free space, inside obstacle, polygon extremal point or boundary. Converting between Cartesian coordinates (x, y) and grid coordinates (i, j) is given by:

$$(i,j) \leftarrow \left(\frac{(x - x_c^o + \frac{s_c^o}{2})}{s_c^o} + \frac{n_{cols}}{2},\ \frac{(y - y_c^o + \frac{s_c^o}{2})}{s_c^o} + \frac{n_{rows}}{2} \right) \quad (1)$$

Converting grid (i, j) to Cartesian coordinates (x, y) is given by:

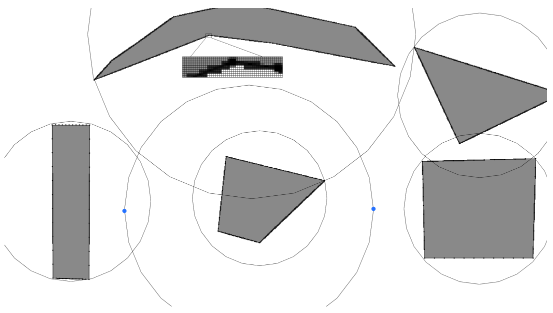
$$(x, y) \leftarrow \left(s_c^o(i - \tfrac{n_{cols}}{2}) + x_c^o,\ s_c^o(j - \tfrac{n_{rows}}{2}) + y_c^o \right) \quad (2)$$

Verifying if an edge intersects an obstacle in a 2D grid requires only the use of a line algorithm such as Bresenham's [17] to check transitions between labels in the grid (e.g., free space to inside obstacle). Other verifications such as if a terminal node is inside an obstacle, requires only the access of a cell in the map and checking the label using (1).

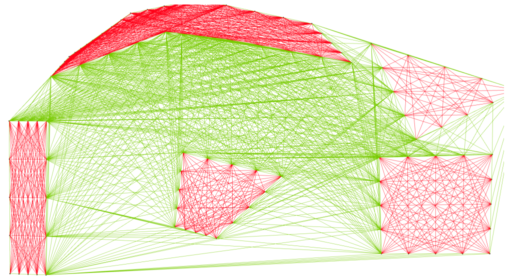
All connections in Fig. 2b correspond to inner and inter-obstacle connections which are viable candidates for future use in the case of having more than two terminal nodes. Another improvement was the use of bounding circumferences (see Fig. 2a). Bounding circumferences (N_o, $o = 1, \ldots, |\mathcal{O}|$) are disks centered in the geometric centroid p_c^o of a soft obstacle and with radius equal to $r^o = \max |p_c^o - p_k^o|$. For two nodes ($p_A$ and p_B) a bounding circumference $N_{\overline{AB}}$ is also computed, centered in $p_{\overline{AB}} = \frac{p_A + p_B}{2}$ and with radius $r_{\overline{AB}} = \alpha \frac{|p_A - p_B|}{2}$, where α is an adaptable gain. Obstacles for which the bounding circumferences N_o are not overlapping $N_{\overline{AB}}$ are discarded from the shortest path computation. The overlapping condition is given by $|p_c^o - p_{\overline{AB}}| \leq r^o + r_{\overline{AB}}$. Note that the adjustment of α gives the opportunity to consider areas with lower costs that might be of interest to the final solution. Figures 2c and 2d denote the shortest paths between the terminal nodes and surrounding obstacles considered for the final solution obtained without and with the bounding circumferences, respectively. With the bounding circumferences, the overall number of connections between the nodes and the obstacles is reduced. In particular, in scenarios containing multiple obstacles far from the two nodes of interest, are completely discarded which in turn, reduces the computation complexity of the shortest path (smaller number of connections).

B. 3-points problem

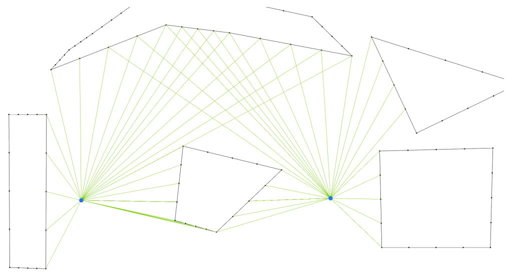
The appearance of a Steiner point results from solving the 3-points problem. Given three points the goal is to compute an additional point that minimizes the sum of shortest path distances to the given points (see Fig. 3). Without obstacles the solution is to solve the Fermat-Torricelli problem [6] (Algorithm 1). For solid obstacles, a solution that builds on the solution without obstacles, and avoiding the obstacles is presented in Algorithm 2. For three points that are not inside an obstacle, a heuristic solution is given by the nearest



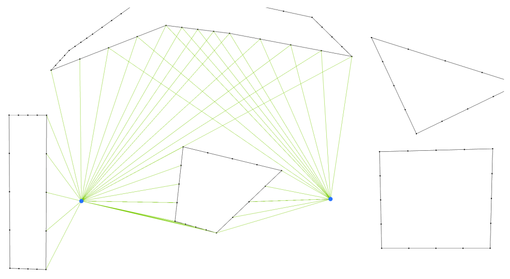
(a) Bounding circumferences computed for each obstacle and for the two terminal nodes.



(b) Representation of the connections inside and outside of the obstacles without considering the terminal nodes.



(c) Representation of the connections to the terminal nodes without bounding circumferences.



(d) Representation of the connections to the terminal nodes with bounding circumferences.

Fig. 2. Scenario with five soft obstacles with polygonal and grid representations and two terminal nodes.

intersection between a line segment and the solid obstacles (**intersectsObstacle**). The line segment is defined by the point that corresponds to the triangle vertex adjacent to the internal angle that gave rise to the Steiner point (computed without obstacles) and that Steiner point. The solution can be further

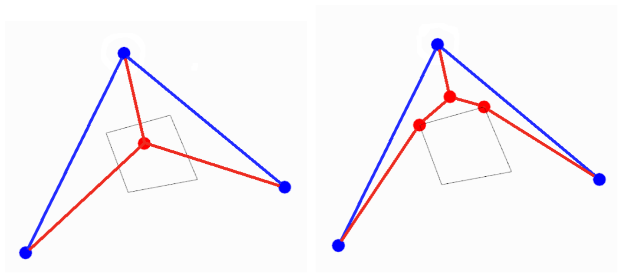


Fig. 3. Simulation results depicting the problem where a Steiner node is created inside an obstacle when the obstacle is ignored and the obstacle is avoided.

Algorithm 1: Exact algorithm to compute the Steiner point without obstacles – $ESteiner(p_A,p_B,p_C)$.

```
Input: Input Nodes ($p_A$,$p_B$,$p_C$)
Output: Steiner point ($p_S$)
1  $\theta \leftarrow \angle(p_A, p_B, p_C)$;
2  $\alpha_1 \leftarrow \angle(p_A, p_B)$, $d_1 \leftarrow |p_A - p_B|$;
3  $\alpha_2 \leftarrow \angle(p_B, p_C)$, $d_2 \leftarrow |p_B - p_C|$;
4  $s \leftarrow -sign(\theta)$;
5  $p_1 \leftarrow (x_B + d_1\cos(\alpha_1 + s\frac{\pi}{3} - \pi), (y_B + d_1\sin(\alpha_1 + s\frac{\pi}{3} - \pi))$;
6  $p_2 \leftarrow (x_B + d_2\cos(\alpha_2 - s\frac{\pi}{3}), (y_B + d_2\sin(\alpha_2 - s\frac{\pi}{3}))$;
7  $m_1, b_1 \leftarrow (\frac{y_C - y_1}{x_C - x_1}, y_1 - m_1x_1)$;
8  $m_2, b_2 \leftarrow (\frac{y_A - y_2}{x_A - x_2}, y_2 - m_2x_2)$;
9  if $|x_A - x_2| = 0$ then
10     $p_S \leftarrow (x_2, m_1x_2 + b_1)$;
11 else if $|x_C - x_1| = 0$ then
12     $p_S \leftarrow (x_1, m_2x_1 + b_2)$;
13 else
14     $p_S \leftarrow (\frac{-b_1+b_2}{m_1-m_2}, m_1x_S + b_1)$;
```

refined by computing the shortest path between the three initial points of the problem and the new Steiner point. This approach generates in general more points (related to new paths defined by the extreme points of obstacles), and transforms the 3-points problem into a local minimum Steiner tree. In order to obtain a solution such as the one presented in Fig. 3 (on the right) at least two more iterations of the 3-points problem are performed.

The problem to obtain the best point to connect 3 points over soft obstacles in order to obtain a minimum cost weighted Euclidean Steiner tree is rather involved. The complexity of this problem is illustrated in Fig. 4. The figure shows the variation of the cost for each point, in the region of interest (discretized), of the 3-points Steiner tree that uses that point as a Steiner point. The figure illustrates the variation of the costs with increasing weight of the soft obstacles, as well as the minimum cost solution for each case (the Steiner point and the corresponding Steiner tree).

To address this problem a heuristic approach is proposed based on a *local search procedure* combined with a *diversification strategy* to avoid local optimal solutions. The local search procedure is focused in a point that is a candidate to be a Steiner point and an intensive search for a better solution

Algorithm 2: Heuristic algorithm to compute the Steiner point with solid obstacles – $HSteinerO(p_A,p_B,p_C,\mathcal{O})$.

```
Input: Input Nodes ($p_A$,$p_B$,$p_C$)
       Obstacles ($\mathcal{O}$)
       Steiner point without obstacles ($p_S$)
Output: Steiner point avoiding obstacles ($p_S$)
1  $collides \leftarrow$ intersectsObstacles($p_B$,$p_S$,$\mathcal{O}$) ;
2  if $collides$ then
3      $p_S \leftarrow$ argmin $(|p_B - (\overline{p_B\ p_S} \bigcap \mathcal{O})|)$;
```

around this point, as can be seen in the local search procedure (Algorithm 3). The input parameter of this procedure S_c, the set of seed nodes, is obtained through a diversification strategy ($Diversification$ procedure in Algorithm 5) that considers a set of Steiner point candidates spread out over the entire region of interest. Three important points are added to this set: 1) the exact solution for 3-points problem without obstacles; 2) the heuristic solution for the 3-points problem avoiding the obstacles and; 3) the point that corresponds to the triangle vertex adjacent to the internal angle that gives rise to the previous Steiner points.

In Algorithm 3, the Δ parameter controls the search of a solution in the neighboring of each candidate c in S_c through a circumference of a radius starting with that value Δ and centered in the candidate ($circumferencePoints$). Each candidate's cost f_c is evaluated by the **WTreeCost** procedure and the best solution (the solution with minimum cost) is stored in p_s. The **WTreeCost** procedure computes the shortest path between the three input points and a candidate point, resulting in three shortest path computations (e.g., for the input points p_A, p_B and p_C and a candidate p_s, the cost is given as $|SP(p_A, p_s)| + |SP(p_s, p_B)| + |SP(p_s, p_C)|$ where SP stands for Shortest Path and the module operator $|.|$ means a weighted Euclidean distance).

For a given number of iterations K or until no more explorations can be performed, for each candidate c in S_c, a set of neighbor candidates C (**circumferencePoints**) is computed. Each candidate in C has a Δ_c distance associated, and for each new candidate e in C a new cost f_e is obtained. If the cost of the candidate e is smaller than c the value is updated, otherwise if none of the candidates in C provides a better solution, the Δ_c is decreased by a λ value, intensifying the search of a solution around this candidate. The Δ_c value is decreased until it is less than a global threshold Δ_{min}. When a candidate reaches Δ_{min} it is removed from C as it means that this region of the solutions space is not going to be further explored. If the new cost is smaller than the cost in p_s the best candidate is also updated. In the examples shown in Fig. 4 the solutions obtained by the proposed approach (using 5 seed nodes and 20 iterations) are the best solutions represented in the figure (black dot).

C. Steiner Tree heuristic

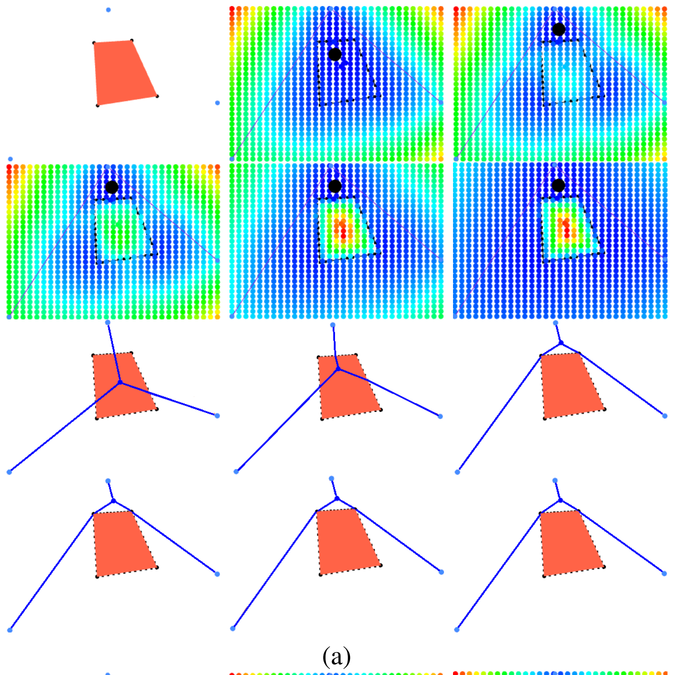
As shown in the previous subsection, computing the 3-points problem in the presence of soft obstacles is a difficult

Algorithm 3: Local search procedure to compute the Steiner point with soft obstacles – $3PointsSO(p_A, p_B, p_C, \mathcal{O}, S_c, K)$.

```
Input: Input Nodes ($p_A$, $p_B$, $p_C$), Soft Obstacles ($\mathcal{O}$)
       Seed Nodes ($S_c$), Max Iterations ($K$)
Output: Steiner Point ($p_s$) and cost $f_{p_s}$
1  $p_s \leftarrow \underset{c \in S_c}{\text{argmin}}\ (\text{WTreeCost}(c, \mathcal{O}, p_A, p_B, p_C))$;
2  for $i = 0$ to $K$ do
3      foreach $c \in S_c$ do
4          $C \leftarrow$ circumferencePoints($c$,$\Delta_c$);
5          $update \leftarrow$ false;
6          foreach $e \in C$ do
7              $f_e \leftarrow$ WTreeCost($e$,$\mathcal{O}$,$p_A$, $p_B$, $p_C$);
8              if $f_e < f_c$ then
9                  $S_c \leftarrow S_c \setminus c$;
10                 $S_c \leftarrow S_c \bigcup e$;
11                 $c \leftarrow e$;
12                 $f_c \leftarrow f_e$;
13                 $update \leftarrow$ true;
14             if $f_e < f_{p_s}$ then
15                 $p_s \leftarrow e$;
16         if $\neg$ $update$ then
17             $\Delta_c \leftarrow \Delta_c - \lambda$;
18             if $\Delta_c \leq \Delta_{min}$ then
19                 $S_c \leftarrow S_c \setminus c$;
```

task, which means that assuming that the Euclidean Steiner tree is generated by a combination of multiple 3-points problems, when some of them may even be codependent (two independent solutions of the 3-points problem may connect to each other, and may need to be iteratively adjusted), makes this problem even more complex. To reduce some of the complexity we compute an initial MST that must be refined in order to reduce the overall cost of the final solution. The proposed algorithm for the computation of the Euclidean minimum Steiner tree considering soft obstacles is divided in two steps. In the first step we generate a MST (Algorithm 4) with soft obstacles (MSTO) using Prim's Algorithm [18]. In the second step an iterative process is applied to the MSTO in order to obtain an ESMT-SO (Algorithm 5).

In the first step, the algorithm starts with the computation of a triangular matrix containing the Euclidean distances between the terminal nodes (**InitializeNodeDistanceMatrix**). Then for each pair of nodes, the overlap is computed between the line segment defined by the nodes and the obstacles (**overlapDetection**). If a collision is detected, a new path is computed that connects the two nodes using the previously presented shortest path algorithm (**ShortestPath_SoftO**). The path is stored to be used later and the distance is updated in the triangular matrix. Prim's algorithm is then used on the triangular matrix to compute the MST considering soft obstacles (**MSTO**).


(a)

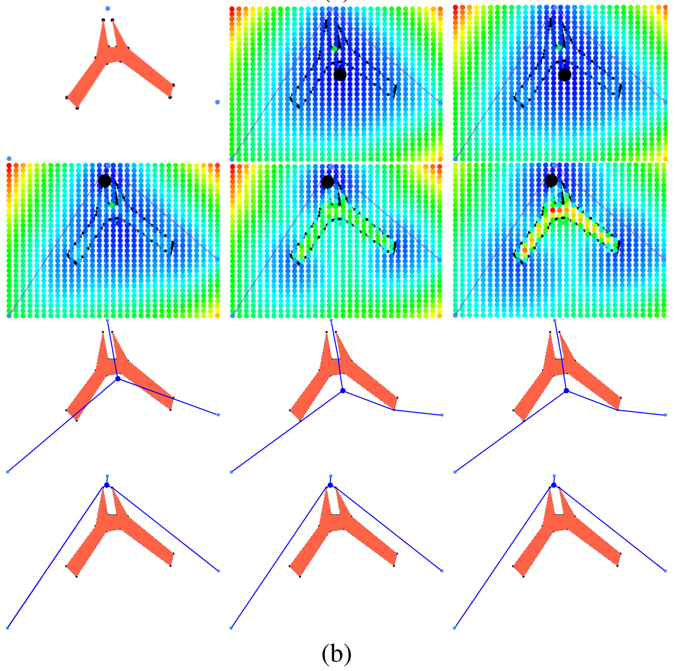

(b)

Fig. 4. 3-points problem with soft obstacles - The first column of the first row of (a) and (b) presents the three points and a soft obstacle. Each point in sub-figures in columns 2 and 3 of row 1 and in row 2 of (a) and (b) corresponds to a Steiner point for three terminal points and a soft obstacle. Smaller costs are represented in blue, higher costs in red. The black dot corresponds to the Steiner point with minimum cost. Rows 3 and 4 of (a) and (b) correspond to the minimum cost solution (in blue). The obstacle costs are (increasing from left to right and from top to down): 1, 1.1, 1.5, 2, 5 and 10.

For the second step, and for each node in the MSTO, the algorithm starts by checking if a node is valid or not (**isValid**). Conditions to be valid include the existence of one or more child nodes. The first step is to compute the minimum angle

Algorithm 4: Algorithm to obtain a minimum spanning tree considering soft obstacles – $MSTO(T, \mathcal{O})$.

```
Input: Terminal Nodes ($T$); Soft Obstacles ($\mathcal{O}$)
Output: MST considering soft obstacles ($MSTO$)
1 $M \leftarrow$ InitializeNodeDistanceMatrix();
2 foreach $(p_i, p_j) \in T$ do
3     $collides \leftarrow$ overlapDetection($(p_i, p_j)$,$\mathcal{O}$);
4     if $collides$ then
5         $L \leftarrow$ ShortestPath_SoftO($(p_i, p_j)$,$\mathcal{O}$);
6         $M(i, j) \leftarrow |L|$;
7 MSTO $\leftarrow$ Prim($M$)
```

Algorithm 5: Algorithm to obtain an heuristic Steiner tree considering soft obstacles.

```
Input: Terminal Nodes ($T$), Soft Obstacles ($\mathcal{O}$) , Max
       Iterations ($K$)
Output: Minimum Steiner Tree with obstacles ($SMTO$)
/* Initialization: Compute all the
   obstacles' connections                    */
1 $MSTO \leftarrow$ MSTO($T$,$O$);
2 $SMTO \leftarrow MSTO$;
3 while $MSTO \neq \emptyset$ do
4     $p_i \leftarrow MSTO(i)$;
5     $MSTO \leftarrow MSTO \setminus \{p_i\}$;
6     if $\neg isValid(p_i)$ then
7         continue;
8     $\theta$,$p_A$,$p_B$,$p_C \leftarrow \min_{j=1:|C(p_i)|}$
        $(\angle(P(p_i) \cup C_{j+1}(p_i), p_i, C_j(p_i)))$;
9     if $\theta < 120°$ then
10        $S_c \leftarrow \emptyset$;
11        $S_c \leftarrow S_c \bigcup ESteiner$ ($p_A$,$p_B$,$p_C$);
12        $S_c \leftarrow S_c \bigcup HSteinerO$ ($p_A$,$p_B$,$p_C$,$\mathcal{O}$);
13        $S_c \leftarrow S_c \bigcup Diversification$ ($S_c$,$p_A$,$p_B$,$p_C$);
14        $f_{old} \leftarrow |SP(p_A$,$p_B)$+$SP(p_B$,$p_C)|$;
15        $p_s, f_{p_s} \leftarrow 3PointsSO(p_A$,$p_B$,$p_C$,$\mathcal{O}$,$S_c$,$K)$;
16        if $f_{p_s} < f_{old}$ then
17            $L_1 \leftarrow$ ShortestPath_SoftO($p_A$,$p_s$,$\mathcal{O}$);
18            $L_2 \leftarrow$ ShortestPath_SoftO($p_B$,$p_s$,$\mathcal{O}$);
19            $L_3 \leftarrow$ ShortestPath_SoftO($p_C$,$p_s$,$\mathcal{O}$);
20            $SMTO \leftarrow SMTO \setminus (p_A$,$p_B$,$p_C)$;
21            $SMTO \leftarrow SMTO \bigcup (L_1$,$L_2$,$L_3)$;
```

between each triplet (parent node ($P()$), actual node and child node($C()$)) or (child node, actual node and child node) in the tree. If the angle is smaller than 120° then, Steiner points are computed as in Algorithms 1 and 2 and added to the seed set S_c, the $Diversification$ procedure is called in order to find more seeds to be added to S_c. Given the triplet (p_A, p_B, p_C), the seed S_c and a set of soft obstacles, the 3-points problem is computed using the Algorithm 3. If the cost of p_s is smaller than the initial cost the shortest paths L_1, L_2, L_3 are inserted or updated in the tree.

III. Validation Results

In order to validate the proposed approach to compute a low-cost Euclidean Steiner tree with soft obstacles, two tests were considered. In the first test some scenarios from [8] (see Fig. 5) were considered where the costs associated with each obstacle represent insurmountable regions. The goal was to compare the proposed solution to the exact one in order to assess how close the proposed heuristic solution is to the exact solution when obstacles are avoided.

The second test (see Fig. 6) uses real data in order to design the spine of a network from scratch in order to connect a set of terminal nodes that represents communication equipment, avoiding disaster prone areas, through a minimum cost infrastructure. Disasters may occur at any time, so in order to guarantee the availability of the network a solution must be able to reduce the risk of being affected by them. Some geographical areas are more prone to natural disasters than others, in particular to the occurrence of hurricanes, floods or earthquakes that to some degree can negatively impact a communication network. For many types of disasters, models may predict or assign a probability to the occurrence of disasters for a given amount of time, based on the previous history of events.

In [19], the 2013 European Seismic Hazard Model was presented. For the validation of the proposed algorithms a simple polygonal overlay of one of the models available in [19] was done - see Fig. 18 "PGA exceedance probabilities of 2 % in 50 years". PGA (Peak Ground Acceleration) is one of the most common parameters used for engineering design purposes. The idea is not to provide an optimal solution to the problem but to present a near realistic scenario. Also, to generate a Euclidean Steiner tree in a realistic network, the COST266 network [20] was adapted for the validation scenario. At the end of this subsection we also present results for the scenario in Fig. 1 for four different costs.

The results for the first test are shown in Table I, which presents values for the weighted Euclidean length of the exact and heuristic solution, as well as the weighted Euclidean length for the MST. The error between the exact and the heuristic solutions is also presented. The results show that in more realistic scenarios, the error is small (e.g., $\leq$ 1% for Scenarios 1-3) where for other scenarios it can go up to 4%. The error of 10% is caused by the initial configuration of the MST that makes it impossible to further improve the tree. For the second test (see Fig. 6), the top figure represents the MST, while the bottom figure the proposed heuristic. This represents a more realistic scenario, as it is possible to notice some interesting connections in the tree. When dealing with smaller weight (e.g., border between Spain and France) the connection follows the obstacle until the cost to traverse the obstacle is minimal. On the other end (e.g., high cost obstacle over Greece), the connection almost always forces the avoidance of the high cost obstacles.

Simulations for the scenario presented in Figs. 1 and 2 using the proposed approach with multiple costs for a random

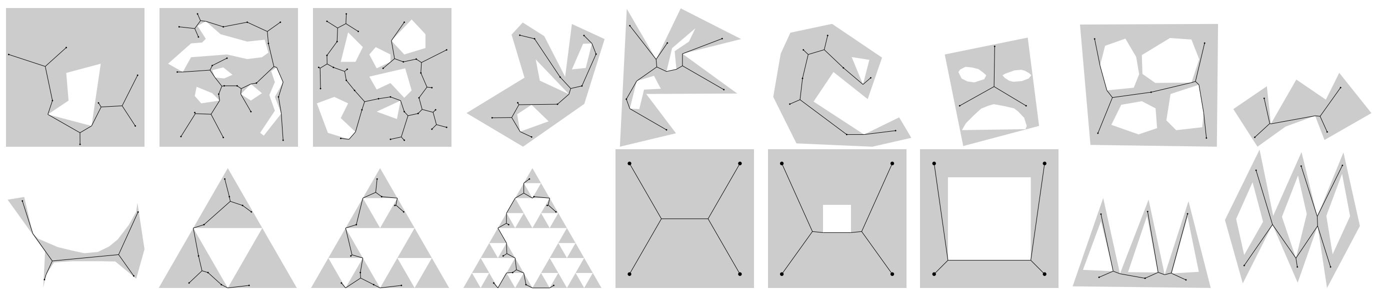


Fig. 5. Scenarios adapted from [8], white areas represent obstacles. The tree depicted in each scenario corresponds to the exact Euclidean Steiner Tree. From left to right and upper to bottom, scenarios 1 to 18 are defined.

TABLE I
RESULTS FOR THE PROPOSED APPROACH APPLIED TO SOME SCENARIOS OF THE DATASET USED IN [8] (SEE FIG. 5).

Exact	Method	Error %	MST	Scenario
1.23162	1.23238	0.06	1.3393	1
5.47342	5.52994	1.03	5.7883	2
5.93503	5.94813	0.22	6.21489	3
15.3413	15.342	0.00	16.0679	4
11.5136	11.588	0.65	12.8935	5
15.9702	15.9725	0.01	16.3397	6
6.00962	6.00994	0.01	6.94067	7
15.7772	17.3823	10.17	17.8215	8
1.11558	1.13176	1.45	1.27075	9
12.3587	12.8664	4.11	13.7153	10
15.0233	15.0233	0.00	15.1859	11
17.093	17.4281	1.96	17.6686	12
18.1115	18.1115	0.00	18.6193	13
2.18564	2.19636	0.49	2.4	14
2.19569	2.19687	0.05	2.4	15
2.29706	2.29706	0.00	2.4	16
14.1726	14.7803	4.29	16.0864	17
17.0214	17.4109	2.29	23.49	18

number of terminal nodes are presented in Fig. 7. From a qualitative analysis of the results it is possible to see that when increasing the weight, the obtained tree tends to find the nearest exit in order to avoid the costs associated with the obstacles.

IV. CONCLUSION

In this paper we presented a novel approach to compute a Euclidean Steiner tree with soft obstacles. The problem is a generalization of the obstacle-avoiding Euclidean Steiner tree problem. We described solutions for the shortest path problem with heuristic modifications to speed up the computation process, a heuristic/meta-heuristic to solve the 3-points problem and a heuristic approach to compute the aforementioned tree. In particular, we highlighted in Fig. 4 the difficulty of computing an exact solution for the 3-points problem due to the influence of the soft obstacles. Presented evaluation tests show the flexibility of the proposed approach and outline future directions of development and research. In particular, the introduction of different polygons to represent existing infrastructure such as highways and bridges that may decrease the overall cost of a solution. Also the introduction of additional costs for connections over the sea may be of interest to better reflect a real communication network. As a final remark it is worth to mention that in this paper we considered soft obstacles as disaster prone areas, but the presented approach can be applied in different fields. Furthermore, this problem can also be considered as a bi-criteria problem where the two objectives are the weighted and the unweighted Euclidean distance. In this case obtaining a set of Pareto's optimal solutions (or an approximation to this set) is an even more challenging problem.

ACKNOWLEDGMENT

Luis Garrote was supported by the Portuguese Foundation for Science and Technology (FCT) under the PhD grant with reference SFRH/BD/88459/2012.

Lúcia Martins has been supported by FCT (Fundação para a Ciência e a Tecnologia) under project grant UID/Multi/00308/2019 and by FEDER Funds and National Funds through FCT under the project CENTRO-01-0145-FEDER-029312.

This article is based upon work from COST Action CA1512 ("Resilient communication services protecting end-user applications from disaster based failures - RECODIS") supported by COST (European Cooperation in Science and Technology).

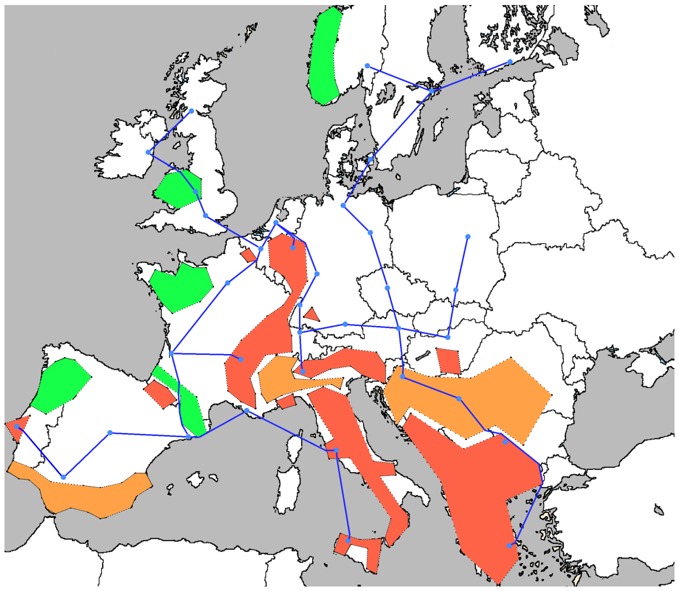
REFERENCES

[1] A. Alashaikh, T. Gomes, and D. Tipper, "The spine concept for improving network availability," *Comput. Netw.*, vol. 82, no. C, pp. 4–19, May 2015.

[2] A. de Sousa, T. Gomes, R. Girão-Silva, and L. Martins, "Minimizing the network availability upgrade cost with geodiversity guarantees," in *2017 9th International Workshop on Resilient Networks Design and Modeling (RNDM)*, Sept 2017, pp. 1–8.

[3] Y. Cheng, D. Medhi, and J. P. G. Sterbenz, "Geodiverse routing with path delay and skew requirement under area-based challenges," *Networks*, vol. 66, no. 4, pp. 335–346, 2015.

[4] A. de Sousa, D. Santos, and P. Monteiro, "Determination of the minimum cost pair of d-geodiverse paths," in *The 2017 International Conference on Design of Reliable Communication Networks (DRCN 2017*, Munich, March 8-10 2017.


(a) Minimum Spanning Tree.

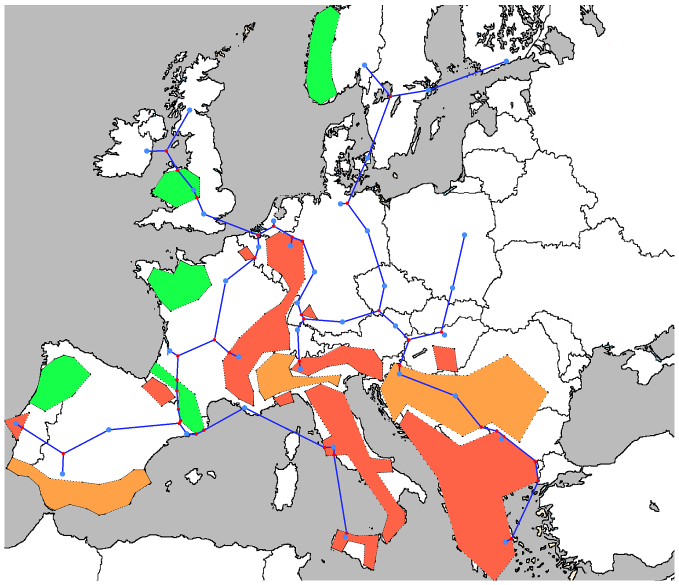

(b) Euclidean Minimum Steiner Tree.

Fig. 6. Representation of the COST 266 network [20] with soft obstacles derived from the seismic hazard model [19]. Light green soft obstacles represent the weight of 1.5, orange soft obstacles with weight 3 and red soft obstacles the weight of 5.

[5] T. Gomes, J. Tapolcai, C. Esposito, D. Hutchison, F. Kuipers, J. Rak, A. de Sousa, A. Iossifides, R. Travanca, J. André, L. Jorge, L. Martins, P. O. Ugalde, A. Pašić, D. Pezaros, S. Jouet, S. Secci, and M. Tornatore, "A survey of strategies for communication networks to protect against large-scale natural disasters," in *2016 8th International Workshop on Resilient Networks Design and Modeling (RNDM)*, Sept 2016, pp. 11–22.

[6] M. Brazil and M. Zachariasen, *Optimal Interconnection Trees in the Plane: Theory, Algorithms and Applications*, ser. Algorithms and Combinatorics. Springer, 2015.

[7] P. Winter and M. Zachariasen, "Euclidean steiner minimum trees: An improved exact algorithm," *NETWORKS*, vol. 30, pp. 149–166, 1997.

[8] M. Zachariasen and P. Winter, *Obstacle-Avoiding Euclidean Steiner Trees in the Plane: An Exact Algorithm*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 286–299.

[9] S. S. Savas, M. F. Habib, M. Tornatore, F. Dikbiyik, and B. Mukherjee, "Network adaptability to disaster disruptions by exploiting degraded-service tolerance," *IEEE Communications Magazine*, vol. 52, no. 12, pp. 58–65, December 2014.

[10] F. Dikbiyik, M. Tornatore, and B. Mukherjee, "Minimizing the risk from disaster failures in optical backbone networks," *Journal of Lightwave Technology*, vol. 32, no. 18, pp. 3175–3183, Sep. 2014.

[11] M. Zachariasen, "A catalog of hanan grid problems," *Networks*, vol. 38, no. 2, pp. 76–83, 2001.

[12] I. Frommer, B. L. Golden, and G. Pundoor, "Heuristic methods for solving euclidean non-uniform steiner tree problems," in *Genetic and Evolutionary Computation - GECCO 2004, Genetic and Evolutionary Computation Conference, Seattle, WA, USA, June 26-30, 2004, Proceedings, Part II*, 2004, pp. 392–393.

[13] J. S. Mitchell, "An algorithmic approach to some problems in terrain navigation," *Artificial Intelligence*, vol. 37, no. 1, pp. 171 – 201, 1988.

[14] J. S. B. Mitchell and C. H. Papadimitriou, "The weighted region problem: Finding shortest paths through a weighted planar subdivision," Cornell University Operations Research and Industrial Engineering, Tech. Rep., 1990.

[15] N. C. Rowe and R. Richbourg, "An efficient snell's law method for optimal-path planning across multiple two-dimensional, irregular, homogeneous-cost regions," *The International Journal of Robotics Research*, vol. 9, no. 6, pp. 48–66, 1990.

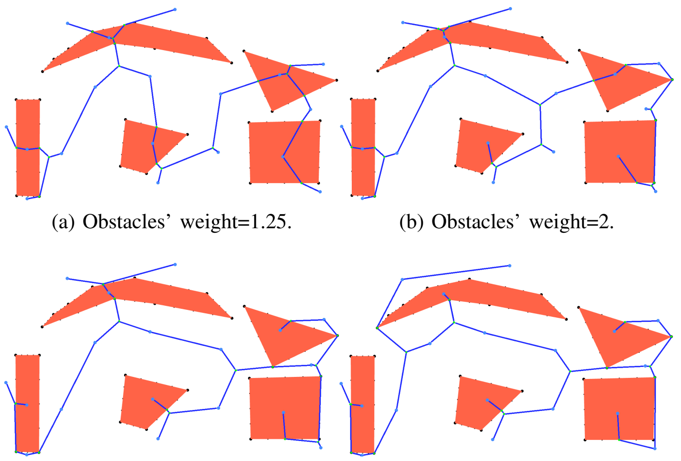
[16] M. Lanthier, A. Maheshwari, and J.-R. Sack, "Approximating weighted shortest paths on polyhedral surfaces," in *Proceedings of the Thirteenth Annual Symposium on Computational Geometry*, ser. SCG '97. New York, NY, USA: ACM, 1997, pp. 485–486.

[17] J. E. Bresenham, "Algorithm for computer control of a digital plotter," *IBM Systems Journal*, vol. 4, no. 1, pp. 25–30, 1965.

[18] R. C. Prim, "Shortest connection networks and some generalizations," *The Bell System Technical Journal*, vol. 36, no. 6, pp. 1389–1401, Nov 1957.

[19] J. Woessner, D. Laurentiu, D. Giardini, H. Crowley, F. Cotton, G. Grünthal, G. Valensise, R. Arvidsson, R. Basili, M. B. Demircioglu, S. Hiemer, C. Meletti, R. W. Musson, A. N. Rovida, K. Sesetyan, and M. Stucchi, "The 2013 european seismic hazard model: key components and results," *Bulletin of Earthquake Engineering*, vol. 13, no. 12, pp. 3553–3596, Dec 2015.

[20] S. Orlowski, R. Wessäly, M. Pioro, and A. Tomaszewski, "Sndlib 1.0—survivable network design library," vol. 55, pp. 276 – 286, 01 2009.


(a) Obstacles' weight=1.25. (b) Obstacles' weight=2.

(c) Obstacles' weight=5. (d) Obstacles' weight=10.

Fig. 7. Euclidean minimum Steiner tree considering soft obstacles with different weights.