Multi-Objective Heuristics For Network Construction In An Obstacle-Dense Environment

Francesco Bernardini[1] and Aaron T. Becker[1]

***Abstract*— We present a heuristic method to construct an optimal communication network in an obstacle-dense environment. A set of immobile terminals must be connected by a network of straight-line edges by adding agents to serve as relays. Obstacles are represented by polygons, unaccessible by the agents of the network or by the edges. The problem with obstacles is reduced to a problem without obstacles by choosing the nodes of the optimal network among the obstacles' vertices that are in mutual line of sight. A second heuristic method is developed to solve the bicriteria optimization problem with number of agents and length of the network as concurrent costs.**

I. Introduction

Optimal network construction is a research topic with applications in several areas of science and technology: notable examples are the design of efficient phone networks among cities, the development of urban sewer layouts, or trace routing in Printed Circuit Board (PCB) design. In all these instances, "optimality" is defined as the minimization of a specific cost function, which depends on the type of network considered and on the constraints imposed by the problem.

The networks we consider are modeled by *weighted graphs*. A weighted graph $G = (V, E, w)$ is a collection of nodes V logically interconnected by edges E that have costs (or "weights") w. If the network has a physical structure then the edges might be the links connecting the physical nodes (e.g., pipes and/or wires) and the weight of the edge might be proportional to the price per unit length/surface/volume of the material used. Conversely, if the nodes of the network are connected by intangible links (e.g. radio links) then the cost can be defined according to the specific technology used for transmission.

Many problems concerning weighted graphs have been presented. One of the most famous examples is the *minimum spanning tree* [8] (MST), where given a graph (referred to in the following as *parent graph*), find the shortest tree that interconnects **all** its nodes. If only a subset $T \subset V$ of the nodes (referred to as *terminals*) of the parent graph requires interconnection, the problem is called the ***Steiner tree** problem for graphs* (STPG) [10] Although conceptually similar, the two problems have different computational overhead: MST can be solved in polynomial time, while the STPG is Non-deterministic Polynomial-time (NP)-hard. Variants of this problem can be defined by a) embedding the terminals in a 2- or 3-dimensional vector space, b) leaving the number and position of additional non-terminal nodes to be determined and c) weighting each edge with some metric. If the metric chosen is the Euclidean distance then the problem is called *Euclidean Steiner tree problem* (ESTP), while if the metric is the L_1 norm (or "taxicab distance") then the problem is denoted as the *rectilinear Steiner tree problem* (RSTP). These problems, in the following referred to as xSTP, are generalizations of the STPG therefore share with it the same complexity class (NP-hard).

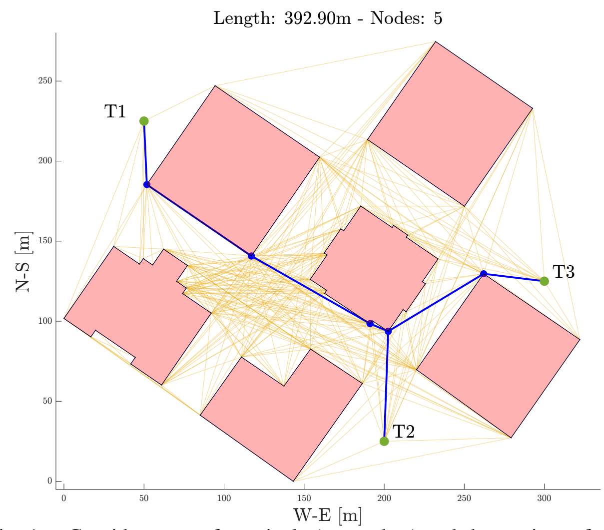

Fig. 1. Consider a set of terminals (green dots) and the vertices of a set of obstacles (pink polygons). A weighted graph is constructed by drawing edges (orange lines) between all pairs of nodes in line of sight, with their Euclidean lengths as weights. From such a graph the Steiner tree is extracted (solid blue lines) and the number of corners used provides an estimate of the number of nodes (blue dots) needed.

In their simplest formulation, ESTP and RSTP place non-terminal nodes everywhere in the environment space. If some regions of the space (polygons in 2D, polyhedrons in 3D) are forbidden then the problem takes the name of **obstacle-avoiding** Euclidean/Rectilinear Steiner tree (OAxSTP). Methods valid for xSTP in general will not solve for OAxSTP and then the problem becomes increasingly difficult with the number of obstacles and/or terminals.

We focus on OA Euclidean STP (OAESTP) where the density of obstacles is much larger than the density of terminals. The main assumption is that in this regime the solution of the OAESTP is well approximated by the solution of a specific STPG, where the parent graph is obtained by connecting all terminals and all vertices of the obstacles

Research sponsored by the Army Research Laboratory and accomplished under Cooperative Agreement Number W911NF-23-2-0014. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. This work was supported by the National Science Foundation under [IIS-1553063, 1932572, 2130793]

[1]University of Houston, Houston, TX USA {fbernardini, atbecker}@uh.edu

which are in line of sight (LoS). Some strategies based on ad-hoc heuristics are further introduced to improve the quality of the solution.

A possible application of this scenario may correspond to the construction of a communication network in a dense urban environment where buildings prevent line of sight (LoS) communication between terminals, forcing the network to adapt its shape around building walls. The nodes are assumed capable of performing highly directional transmissions, as studied by Egarguin et al. in [3], [4], [5], thus the communication links are graphically described by straight lines.

The construction of the network has two constraints: (a) the *length* of the network and (b) the *number of nodes* it involves. The length of the network provides information on the average distance between nodes of the network which corresponds to the cost of communication (see Friis formula [6]). The larger the distance, the higher the power required to transmit a signal within a given signal-to-noise ratio at the receiver (at a fixed frequency/wavelength). Conversely, the number of nodes corresponds to the number of agents required to constitute the network (the cost of the fleet). These two costs are concurrent, since increasing the number of agents and placing them in specific locations in general lowers communication costs. Conversely, decreasing the number of agents increases the average inter-drone distance, and with it the communication cost. The problem is then defined as a multi-objective optimization scheme (a *bi-criteria* optimization). We contribute a solution of a heuristic optimization method that applies Yen's algorithm [21] to the STPG.

Section II reviews some of the existing results in network design with and without obstacles. Section III introduces the model and the methods used. Section IV compares the performance of the procedure presented with existing methods. Section V defines the bicriteria optimization problem and the method used to solve it. Section VI describes the applications of the method to two different urban scenarios. Finally, Sec. VII summarizes the results and outlines possible paths forward for this research.

II. Related publications

The origins of Steiner trees date back to the 17th century [10] with the first studies on the Fermat-Torricelli point in a triangle, which can be considered an instance of the ESTP with three terminals. In the first half of the 20th century this problem was generalized to a higher number of terminals, and eventually named after Jakob Steiner, a Swiss mathematician contemporary of Fermat and Torricelli. Since then a vast amount of literature on the topic has been produced and generalizations to higher-dimensional spaces or manifolds have been presented. A recent review of the state-of-the-art for exact and approximate solving methods can be found in Ljubic [12].

Significantly less literature is available on Steiner trees in the presence of obstacles. Heuristic algorithms, such as that of Armillotta & Mummolo [1], were already known in 1988 while in 1999 Rao [17] described some approximating techniques for Steiner trees introducing the "banyan" structure, a structure which is assumed to be no longer than $(1+\epsilon)$ of the exact Steiner tree length. In 1999 Zachariasen and Winter [22] described the first exact algorithm for the solution of the Euclidean Steiner tree in the presence of obstacles. In 2007 Asadi [2] considered an OAESTP where the terminals are contained *inside* a polygon. In 2010 Muller [13] proposed an improvement in the computation speed of Rao's method [17] by suitably selecting a subset of the obstacles' vertices. In 2021 Parque [15] described a hybrid approach combining hierarchical optimization with gradient-free stochastic optimization. Also in 2021, Rosenberg [19] proposed a genetic algorithm whose evolutionary operators are capable of adapting to the structure of the specific ESTP considered, but considers both *solid* (impenetrable) and *soft* (penetrable at an additional cost) obstacles.

Aside from its theoretical interest, the obstacle avoiding rectilinear Steiner tree problem (OARSTP) has several important practical applications. Hsie [9] defined and solved an OARSTP to improve the layout of an urban sewer network with respect to standard design techniques while Tang [20] provided a comprehensive review of the most recent advances in the routing problem in Very Large Scale Integration (VLSI) design. Garrote [7] determined disaster-aware networks, by assigning a "cost" to different regions of the earth according to their natural disaster likelihood. Such regions, considered as soft obstacles, are then connected by solving an OAESTP which determines the network of minimum risk. Finally, multi-objective communication networks were studied by Levin [11] by means of a composite macro-heuristic that starts with a spanning tree, clusters the nodes of the network, determines a Steiner tree over the clusters, and eventually optimizes that choice.

A search of the relevant literature showed examples of OAxSTP where the number of terminals N_t was comparable or extremely larger than the number of obstacles N_O, while the case $N_O \gg N_t$ was never explored. This work proposes to fill this gap by systematically exploring the case where the number of obstacles is arbitrarily high.

The method we use to build Steiner trees is a simplified version of the concepts expressed in [17] and [13] that makes use only of obstacle vertices. The ϵ of the $(1+\epsilon)-$approximation is argued to be the ratio between a characteristic "small distance" d and the length L of the Steiner tree, possibly proportional to the number of Steiner points (which itself is bounded by N_t).

We are inspired by urban scenarios, where two characteristic distances can be naturally defined: (a) the average street width d_{str} and (b) the average building side d_{bdg}. The urban map, minus the buildings, is then considered as an intrinsic irregular grid obtained as the union of many "tiles" corresponding to approximately rectangular crossroads and alleys of areas:

$$A_{c.r.} = d_{str}^2,\ A_{al} = d_{str} d_{bdg}\ . \tag{1}$$

Assuming $A_{c.r.} \sim A_{al} \equiv \mathcal{A}$, then ϵ can be defined as

$$\epsilon \equiv \frac{\sqrt{\mathcal{A}}}{L}\ . \tag{2}$$

If $\mathcal{A}$ can be assumed constant across a city, ϵ will decrease if the number of terminals is kept constant but they are spread over a larger portion of the urban map.

III. Network construction

The main difference between ESTP and STPG is that ESTP is defined in a continuous space; even in its simplest formulation without obstacles, the solution is found by displacing one or more additional nodes in the ambient space until a configuration of the network with minimum length is found. On the contrary, the structure of a STPG is fundamentally discrete and constrained to the terminals, nodes, and edges that are given as input. Despite this large difference, ultimately both the solutions of an ESTP and of a STPG are a discrete tree. In other words, for the ESTP the cardinality of the ambient space is not reflected in the cardinality of the solutions.

The continuous character of the ESTP allows it to update the list of its nodes and edges during the optimization process. For this reason it will have better or equal solutions than a STPG initialized with the same terminals and a parent graph over which to calculate the Steiner tree. The condition for the equality is that all nodes and edges contained in the solution of the ESTP are a subset of the parent graph of the STPG. This solution is not known a priori, thus an uneducated guess of the parent graph of the STPG will, in general, miss some crucial nodes or edges.

A. Distances

The solutions of a STPG and of an OAESTP can be compared by means of suitably defined “distances”. Although not distances in the mathematical sense, these functions give quantitative information about how much two solutions differ and can point to where improvement is needed. The main requirement is that identical solutions have a distance of zero.

A first estimator can be the fraction of edges in common: let S and S' be two solutions

$$S = (V, E, w),\ S' = (V', E', w')$$
$$D_1(S, S') \equiv 1 - \frac{|E \cap E'|}{\max\{|E|, |E'|\}}\,. \tag{D.1}$$

If the two solutions share the terminals then let A_2 be the area enclosed between them and A_G the area of the convex hull of the graph. A second distance is defined as

$$D_2(S, S) \equiv \frac{A_2}{A_G}\,. \tag{D.2}$$

Finally, if the terminals are displaced between two solutions but their number does not change then a third distance is defined as the area A_3 between the two solutions divided by the area $A_{GG'}$ of the convex hull of the two graphs G, G' that generate the solutions:

$$D_3(S, S') \equiv \frac{A_3}{A_{GG'}}\,. \tag{D.3}$$

The main difference between A_2 and A_3 is that A_2 is defined using only edges already present in the solutions. Conversely, the closed polygons of A_3 are obtained introducing fictitious

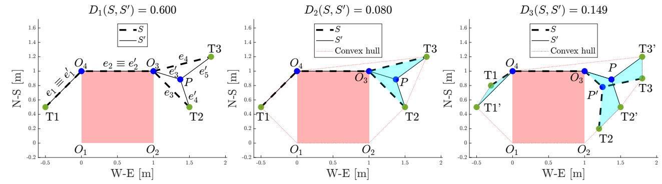

Fig. 2. Graphical representation of the distances of Sec. III-A. D_1, D_2: The two solutions S (dashed) and S' (solid) share terminals and two edges, but depart from each other in proximity of the $n_{\mathrm{d}3}$ point. D_3: S and S' have the same number of terminals but they are displaced. In light blue the area contained between the two solutions, and a dotted line demarcates the convex hull of the points involved.

edges which connect pairs of corresponding terminals. Figure 2 shows a graphical example of the distances.

Distance (D.1) only counts edges which are perfectly overlapping. Therefore even if $S_1 \to S_2$ continuously, then D_1 will reach 0 in finite steps with a discontinuous profile. Conversely, distances (D.2) and (D.3) consider areas so tend to 0 continuously if $S_1 \to S_2$ in the same manner.

Distance (D.1) will find application in Sec. IV where the solutions obtained with different methods will be compared. Distances (D.1) and (D.2) will be used to justify the assumptions of Sec. III-C. Finally, (D.3) is necessary to consider the more general case where terminals are allowed to move, but otherwise is outside of the scope that we are presenting.

B. Classification of nodes

The non-terminal nodes of the solution of an OAESTP and of a STPG are of two types: a) nodes of degree 2 (in the following $n_{\mathrm{d}2}$) obtained by bending around an obstacle and b) Steiner points, i.e. nodes of degree 3 ($n_{\mathrm{d}3}$). Nodes of degree higher than 3 are ruled out in the OAESTP by the requirement that the edges connecting to a Steiner point describe angles of exactly 120°. For the STPG this is not true in general and nodes of higher degree cannot be ruled out a priori. For this reason, a statistical analysis has been performed considering the case of $N_t = [4, 7]$ terminals and randomly initializing their positions for 1000 trials with the configuration of obstacles of Fig. 1. Only one solution containing a n_{d4} was present for $N_t = 4, 5$, 2 for $N_t = 6$ and 7 for $N_t = 7$; no nodes of degree larger than 4 ever occurred. A n_{d4} is never optimal in an ESTP thus its presence implies a lack of additional vertices to form shorter networks with nodes of lower degree and extremely fortuitous placements of terminals. Following this reasoning it can be argued that increasing the density of obstacles (and therefore, the lattice of (2)) while keeping the terminal density low would decrease the likelihood of $n_{\mathrm{d}4}$ or higher, leaving solutions of STPG containing only nodes of degree up to three.

C. The regime

The method we present assumes a regime in which the density of obstacles ρ_O is large compared to the density of terminals ρ_t:

$$\rho_t \equiv \frac{N_t}{A},\ \rho_O \equiv \frac{N_O}{A},\ \rho_O \gg \rho_t\,, \tag{3}$$

where A is the area of the region considered.

From the considerations of Sec. III-B, this choice decreases the likelihood of n_{d4} in the solutions of the STPG,

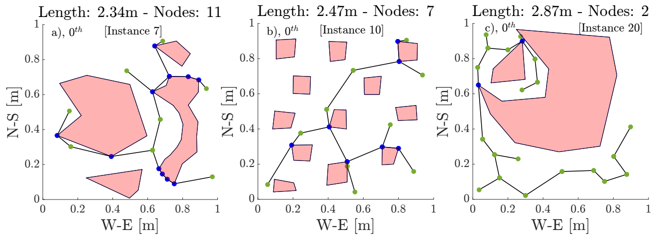

Fig. 3. Solutions of the instances considered in [19] at 0^{th} order of optimization: Instance #7 (a), #10 (b), #20 (c). Green dots are terminals and blue dots are nodes. The quality of the solutions is worse than those of [19] by 1–3%, compatible with the estimate of (5).

rendering the type of nodes that can be encountered homogeneous with those of the OAESTP. Furthermore if terminals are outnumbered by obstacles then their likelihood of having mutual LoS will decrease. Thus, the number of edges bending around obstacles (namely, n_{d2}) increases. While the number N_s of Steiner points in an OAESTP is bounded by the number of terminals ($N_s \le N_t - 2$), the number of n_{d2} depends on the number of obstacles touched by the solution, thus increasing N_O while keeping N_t fixed decreases the ratio $N_{n_{d3}}/N_{n_{d2}}$. Finally, a continuous curve may be approximated by a discrete grid up to the grid's step. A higher density of obstacles' vertices constitutes an effective (although irregular) grid which can be considered to approximate the nodes of the Steiner tree up to the average distance between nearest neighbors. In regime (3) the above considerations support assuming the solutions of OAESTP and STPG are close in the sense of (D.1) or (D.2). In particular it is reasonable to expect that the n_{d2} will be captured correctly and that the main differences will be due to a misplacement of the exact Steiner points. This misplacement becomes less and less relevant as the ratio $N_{n_{d3}}/N_{n_{d2}}$ decreases. Let $\delta R_j > 0$ be the increase in the length L of the network due to the j^{th} misplaced n_{d3}, then assuming regime (3) is equivalent to assuming:

$$\frac{\sum_j \delta R_j}{L} \simeq \frac{\delta R_j N_{n_{d3}}}{(N_t + N_{n_{d2}} + N_{n_{d3}})\langle E\rangle + \delta R_j N_{n_{d3}}} \ll 1\,, \tag{4}$$

where $\langle E\rangle$ is the average length of an edge.

An estimate of the error can be obtained by considering three nodes placed on the vertices of an equilateral triangle of side 1. It is a well known fact that connecting the nodes directly takes a network of length 2, while connecting each node to the circumcenter of the triangle takes a network of length $\sqrt{3}$, that is, a relative error $\delta L/L \simeq 13\%$. For each Steiner point approximated to a vertex, the solution obtained by the STPG is expected to increase the network cost by

$$\delta R_j \simeq 0.13\langle E\rangle\ . \tag{5}$$

This also implies that for N_t terminals this error cannot be larger than $0.13(N_t - 2)\langle E\rangle$.

D. Building the graph

With the same notation of Sec. III, let $\mathcal{O}_v$ be the set of all obstacles' vertices and $\mathcal{T}$ the set of all terminals. Then $G = (V, E, w)$ is obtained by defining

1) V as the set of all obstacles' vertices and terminals:

$$V \equiv \mathcal{O}_v \cup \mathcal{T}\ . \tag{6}$$

2) E as the set of all edges connecting pairs of elements V without intersecting any obstacle.
3) w as the Euclidean lengths of each edge in E.

Path minimization problems involving obstacles usually consider only edges tangent or bitangent to obstacles. Edges that would pierce the obstacle if they were continued after their endpoints, never belong to any shortest path between two points unless the endpoints themselves belong to some obstacle's contour. However, for trees this is not necessarily true and there may be instances where a piercing edge improves the solution of the STPG (see Fig. 3b). For this reason the method used in the following sections to solve the STPG will build E considering all edges that connect vertices in LoS and will be referred to as *LoS-Vertices* (LoSV).

IV. Performance comparison

Once a parent graph is defined, any solver capable of handling STPG may be used to find the solution. We integrated SCIP-Jack [18] in a custom MATLAB routine to solve the various instances of the problem. As assumed in Sec. III, the method is assumed to give its best results in the regime of (3). To enable a direct comparison we use the same instances with *solid* obstacles tested in [19] using the genetic algorithm method developed therein (called *StOBGA*). In particular we compare the instances numbered as #7, #10 and #20. In these instances the density of terminals is comparable to, or much larger than the density of obstacles, therefore they do not represent regime (3) and are not a setup where method LoSV is expected to outperform method StOBGA. However, the comparison still teaches valuable lessons for future developments, and can represent a lower bound for LoSV's performances.

TABLE I
Results comparison with StOBGA method

Problem	[**7**,8,4]	[**10**,10,12]		[**20**,20,2]
	[**Instance #**, # Terminals, # Obstacles]			
StOBGA	2.31 m	2.4211 m		2.80 m
LoSV (0^{th})	2.34 m	2.4704 m	(0^{th})	2.87 m
		2.4240 m	(1^{st})	
		2.4212 m	(2^{nd})	
%	1.3 %	2.0 %	(0^{th})	2.5 %
		0.1 %	(1^{st})	
		< 0.005 %	(2^{nd})	
$D_1(S_{Lo}, S_{St})$	$1 - 13/21$	$1 - 7/17$	(0^{th})	$1 - 10/28$
		$1 - 14/17$	(1^{st})	
		0	(2^{nd})	

Figure 3 shows the results obtained by LoSV, while Table I compares them to those of StOBGA. As expected, the solutions obtained by LoSV have worse results than StOBGA within 3 %. However, LoSV's solutions to the instances with less terminals and more obstacles, #7 and #10, appear to reproduce more accurately the structure of StOBGA's solutions, i.e. they are closer in terms of distance (D.1) than the solutions of #20, which has 20 terminals and only 2 obstacles. Furthermore, all vertices present in StOBGA's solutions are also used by LoSV's solutions, suggesting the ability of LoSV to predict at least a superset of the n_{d3} of the exact solution.

A. Finding and optimizing the solution

The solution found with method LoSV can be considered as a 0^{th} order approximation of the OAESTP. Several heuristic rules can be applied to the 0^{th} order solution to improve the result. The optimization rules depend on the complexity of the structures that are formed during the optimization of the STPG, which is loosely proportional to the number of vertices of the polygon interested by the optimization (triangle, quadrangles and so on). A first-order approximation is obtained by the following actions:

1) **3-vertices:** A three-point structure where one point is connected to the other two may admit a Steiner point of degree 3 if the angle between the two vertices is less than $120°$. Alternatively, a non-terminal n_{d3} can be considered a 3-terminals proto-Steiner structure and its position can be varied if this improves the cost of the structure and the new location does not fall within an obstacle.
2) **4-vertices:** Four point structures without additional Steiner points may be transformed into full Steiner topologies if this improves the length of the network. Alternatively, two n_{d3} directly connected form a 4-terminals proto-Steiner structure and their position can be varied.
3) **5- or more vertices:** Similar considerations can be put in practice for structures with higher number of vertices, or equivalently, for direct interconnections of more than two n_{d3}.

An application of these concepts can be seen in Fig. 4:

- The 0^{th} order solution, Fig. 4a, determines two proto-Steiner structures, with one and with three n_{d3}.
- A first round of optimization, Fig. 4b, determines the optimal positions of the nodes of both structures assuming no obstacles and if the result does not lie within obstacles then adds them to the graph. The structure might not be optimal yet, and the addition of new nodes and edges can verify this situation. This is the case of the structures with one n_{d3}, while the other with three n_{d3} cannot be optimized further.
- A second round of optimization, Fig. 4c, seeks for improvement on the new structure with one n_{d3}.

These optimization steps are performed using a standard gradient descent procedure. The total gain for this specific instance has been of $\sim 2\,\%$, mostly due to the optimization of the structure with three n_{d3}. The likelihood of encountering structures with higher number of vertices is expected to be low in the regime of (3). The $0.0487\,\mathrm{m}$ improvement is in line with that predicted by (5) assuming an average edge of $\langle E\rangle \simeq 0.1$ and $N_s = 4$ Steiner points:

$$\delta L \simeq 0.13 \langle E\rangle N_s = 0.52\,\mathrm{m}\ . \tag{7}$$

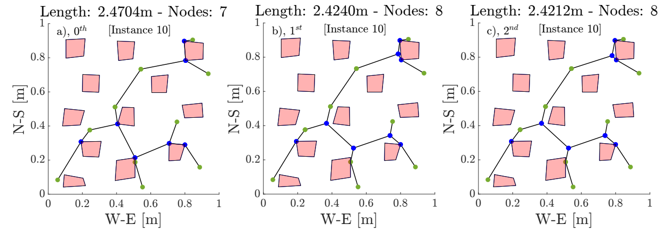

Fig. 4. Application of the heuristics of Sec. IV-A to instance #10 of [19]. The 0^{th} order solution, (a), shows a suboptimal 5-vertices and a 3-vertices structure. In the 1^{st} order solution, (b), these structures are optimized using a gradient descent method. The 3-vertices structure requires addition of a node; therefore it may need further optimization. The 5-vertices structure is already optimal. In the 2^{nd} order solution the 3-vertices structure becomes optimal. The number of nodes required may vary between optimization steps.

The number of nodes needs not be constant during optimization; note that the 2^{nd} order solution uses 8 nodes while the 0^{th} order one uses only 7.

An important feature of the method is that once the map of the obstacles is known then a preliminary parent graph G_0 can be built. In the regime (3) G_0 makes up the most part of the parent graph G, lacking only terminals and the comparatively small set of edges which connect them to the vertices within LoS. If the terminals move then it is sufficient to remove those edges and recompute them with the new positions while leaving G_0 fixed. Furthermore, these rules only depend on the n_{d3} and its nearest neighbors. Thus they can be computed and applied independently by the node itself in a context of *distributed* computation.

V. Multiobjective optimization

The key takeaway from the results of Sec. IV-A and IV is that solutions of different length L have, in general, a different number of nodes N_n. This implies the possibility of manipulating the number of nodes of a network by altering its length and viceversa.

Previously the term "solution" described a tree of minimum length. From now on, the term "solution" will describe a pair $(\tilde{L}, \tilde{N}_n)$, where $\tilde{L}$ is the minimum length attainable within a given graph using exactly $\tilde{N}_n$ non-terminal nodes:

$$\tilde{L} = \min\{L : \text{network uses } \tilde{N}_n \text{ nodes}\}\ . \tag{8}$$

A generic non-optimal network $(L, \tilde{N}_n)$ will have $L \geq \tilde{L}$ for a given $\tilde{N}_n$.

Our main objective is to obtain control over the number of nodes used to build a network. Practical applications often require this type of control. For example: let obstacles and terminals be static and let $N_n(t)$ be the number the number of nodes available at any time t. The question is: can connectivity be ensured if $N_n(t) < N_n(0)$?

Conversely, let terminals or obstacles be dynamic, but let $N_n(t) = N_n(0)$ be constant. The solution of the STPG in general will not have the same structure over time. In particular, the number of nodes required can change. In this case the question is if there exists at least one network $(L, N_n(0))$ among which to pick that has minimum L.

The answer to both questions lies in the solution of the bicriteria optimization problem defined in Sec. I, which provides all possible best solutions $(\tilde{L}, \tilde{N}_n)$ at a given time, also referred to as the Pareto front. A solution belongs to the Pareto front if no solution with better $\tilde{L}$ **and** $\tilde{N}_n$ exists, and if that is the case then the solution is said to be *non-Pareto-dominated*. If the Pareto front contains a solution with $\tilde{N}_n$ equal to the desired one, then the problem is solvable.

A. Yen's algorithm

Let $\bar{L}$ be the solution of the STPG for a given parent graph G. By definition no other network can be shorter than $\bar{L}$ although the number of nodes $\bar{N}_n$ it involves needs not be the minimum possible. So let $(\bar{L}, \bar{N}_n)$ be the corresponding pair in the bicriteria optimization problem.

From a Pareto domination perspective, only consider solutions with $N_n < N_{n,\tilde{L}}$ as otherwise any network (L, N_n) would have

$$L \geq \tilde{L},\ N_n \geq N_{n,\tilde{L}}\,, \tag{9}$$

and would be Pareto-dominated by $(\bar{L}, \bar{N}_n)$. Therefore, the Pareto front of the problem is determined by the set of all networks $(\tilde{L}_{N_n}, N_n)$ such that $N_n < \bar{N}_n$. In other words, the Pareto front is a subset of the set of all *suboptimal* networks, regardless of the number of nodes used. For what concerns the minimum value of N_n for a STPG without obstacles, it would be 0 as all terminals are in line of sight, whereas in presence of obstacles it will depends on their placement relative to the terminals.

If the network only has two terminals, therefore a *path*, then there exists a systematic method to determine all the suboptimal paths that connect the terminals within the parent graph G up to the k_s^th^ order (k_s is a positive integer and $k_s = 1$ corresponds to the optimal path). This method is called *Yen's algorithm*.

B. The LoSV-Yen method

Although a version of Yen's algorithm is not known to exist for trees, it is possible to craft a heuristic version that suits our needs. Since it is a combination of LoSV and Yen's algorithm the method will be referred to as method **LoSV-Yen**. The first step of LoSV-Yen is to divide the original network $(\bar{L}, \bar{N}_n)$ in *branches*. A branch is defined as a path on a graph whose endpoints are either terminals or n_{d3}. The second step is to define the list of operations that can be performed on the original network $(\bar{L}, \bar{N}_n)$ or on the individual branches:

1) Apply Yen's algorithm to one of the branches up to the k^{th} order.
2) Displace one of the n_{d3} from its current vertex to another vertex.
3) Modify the topology of the network by removing or adding n_{d3} and/or edges.

Each application of 1) will generate k different branches while each application of 2) will produce just one different network. Therefore applying 1) to a network with p branches together with q displacements of n_{d3} will give rise to qk^p new networks.

The effect of 3) is harder to predict. However, if the solutions are required to be close in the sense of the distance

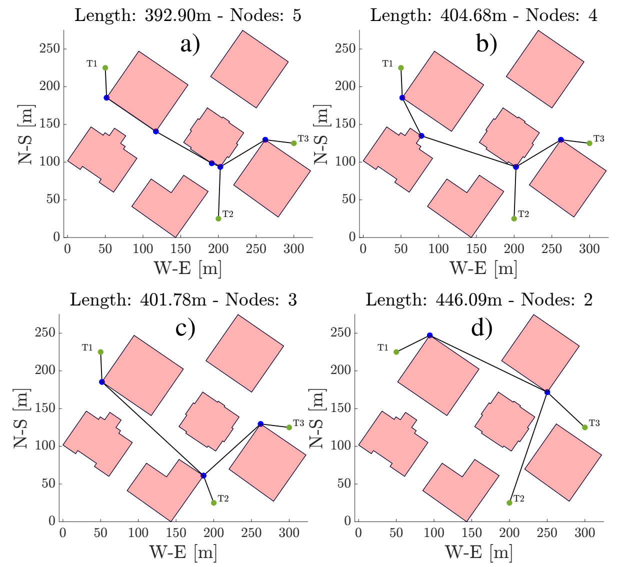

Fig. 5. Application of method LoSV-Yen to an instance with three terminals and 6 obstacles: the original network (a) has three branches and one n_{d3}, and can be altered either applying Yen's algorithm to the branches (b) or by displacing the n_{d3} (c-d). Branches connected to T1 and T2 are not varied as they have less than two nodes, so Yen's algorithm has been applied to the branch connecting to T1 using $k = 30$.

defined in (D.2) then this operation can be ruled out as well as large branch alterations/node displacements. The closeness requirement may arise if a *third optimization criterion* is introduced such as the cost of displacing a node. Under this assumption the operations on the network we analyzed will be limited to just 1) and 2).

Among the new networks created by applying LoSV-Yen, all those with $N_n \geq \bar{N}_n$ will be discarded. Among the remaining, those with minimum length will constitute the approximate Pareto front.

For a finite k and a finite number of displacements only a fraction of all possible configurations will be explored, thus the Pareto front obtained in this manner can only be partial. However, increasing k and displacing the n_{d3} far from their initial points generates networks which are larger and will likely have a larger number of nodes, therefore they would not contribute to the Pareto front anyway.

A small configuration with $N_t = 3$ terminals and $N_O = 6$ obstacles is considered in Fig. 5. The solution determined by the solver for the STPG (a) can be decomposed in three branches with 5 nodes of which one is a n_{d3}. Branches with 0 or 1 nodes cannot improve the number of nodes without displacing one of the endpoints and thus are not varied.

VI. Applications

In this section method LoSV is applied to instances with a larger number of obstacles. A large database of data to test the algorithm is provided by geomapping companies. The data used in the present work was retrieved from the website OpenStreetMap [14] and manipulated with the geomapping software QGIS® [16] to prepare it for use within MATLAB®. The plots shown in Fig. 6 represent a portion of downtown Houston, Texas along with the networks generated by method LoSV-Yen for the case of $N_t = 5$ terminals.

The simulations showed that a plain application of Yen's algorithm to larger graphs ($\sim$400 obstacles, $\sim$2000 nodes and $\sim$28000 edges) is slow in providing results that differ significantly from the initial ones. The reason is that with more nodes there are more paths with comparable lengths that can be built in a neighborhood of the original path. Paths with less nodes, but a significantly larger length, may be never visited by the algorithm if the $k_s^{(i,j)}$ chosen for the i^{th} branch and for the j^{th} node displacement is not high enough. This obviously increases the computation time.

However, most of the variations found by Yen's algorithm have a larger number of nodes and are discarded, and the procedure did not yield any meaningful new configuration ($N_n < \bar{N}_n$) even for $\sum_j \prod_i k_s^{(i,j)} \sim 10^9$.

To overcome the problem and obtain meaningful answers in a shorter time, a slight customization of Yen's algorithm has been applied. At each iteration of Yen's algorithm the first edge of the path is completely removed from the graph. This results in subsequent paths being pushed to perform more significant changes to their initial direction. This modification yielded 10^2 results with $N_n < \bar{N}_n$ within a number of iterations $\sum_j \prod_i k_s^{(i,j)} \sim 10^7$ and the outcome is shown in Fig. 6. The networks have the same structure as only operations 1) and 2) from Sec. V-A were applied.

VII. Conclusions and path forward

The first part of our work focused on the OAESTP in presence of an arbitrarily large density of obstacles and a correspondingly lower density of terminals. Method LoSV was proposed which approximates the solution through a STPG using only obstacles' vertices. The error obtained by this solution is estimated to depend on the number of n_{d3} of the exact solution and therefore bounded by the number of terminals N_t. Such error can be tolerated if comparable with or lower than other intrinsic errors of the chosen application or can be mitigated by applying heuristic rules to specific subsets of the solution of the STPG. Such heuristic rules depend on local properties of the graph and are suitable for a context of distributed computation.

The second part of our work deals with the bicriteria optimization problem of a network where length and number of nodes are concurrent costs. An approximate Pareto front of solutions $(\tilde{L}, \tilde{N}_n)$ is determined by applying method LoSV-Yen, a customized version of Yen's algorithm specialized to trees, to the shortest solution $(\bar{L}, \bar{N}_n)$. We represent the first attempt to apply Yen's algorithm to Steiner trees and hopefully this will stimulate more formal approaches to the problem in the community.

Future paths forward of this research include allowing terminals to move and nodes to be placed over obstacles at an additional cost (*soft* obstacles). As the largest part of the computation is represented by the construction of the preliminary parent graph G_0, further effort will be directed to the construction of efficient algorithms to test LoS between pairs of vertices. As geomapping companies provide a virtually unlimited pool of data, it is of interest to test the method with different urban morphologies and to create a common database of instances to compare the results with other methods. Finally, as mentioned in Sec. V-A, the problem can be expanded to a three-criteria optimization by introducing the cost of moving nodes among configurations determined by LoSV-Yen.

References

[1] A. Armillotta and G. Mummolo, "A heuristic algorithm for the Steiner problem with obstacles," in *Tech. Rept.* Dipt. di Progettazione e Produzione Industriale, Univ. degli Studi di Bari Bari, 1988.

[2] A. Asadi and M. R. Razzazi, "Euclidean Steiner minimal tree inside simple polygon avoiding obstacles," in *2007 International Conference on Computational Science and its Applications (ICCSA 2007)*. IEEE, 2007, pp. 201–207.

[3] N. J. Egarguin, D. R. Jackson, D. Onofrei, J. Leclerc, and A. Becker, "Adaptive beamforming using scattering from a drone swarm," in *2020 IEEE Texas Symposium on Wireless and Microwave Circuits and Systems (WMCS)*. IEEE, 2020, pp. 1–6.

[4] ——, "Beam focusing by scattering from an array of scatterers on a drone," in *2021 United States National Committee of URSI National Radio Science Meeting (USNC-URSI NRSM)*. IEEE, 2021, pp. 140–141.

[5] ——, "Beam focusing by using scattering from drones," in *2021 IEEE 19th International Symposium on Antenna Technology and Applied Electromagnetics (ANTEM)*. IEEE, 2021, pp. 1–1.

[6] H. T. Friis, "A note on a simple transmission formula," *Proceedings of the IRE*, vol. 34, no. 5, pp. 254–256, 1946.

[7] L. Garrote, L. Martins, U. J. Nunes, and M. Zachariasen, "Weighted euclidean steiner trees for disaster-aware network design," in *2019 15th International Conference on the Design of Reliable Communication Networks (DRCN)*. IEEE, 2019, pp. 138–145.

[8] R. L. Graham and P. Hell, "On the history of the minimum spanning tree problem," *Annals of the History of Computing*, vol. 7, no. 1, pp. 43–57, 1985.

[9] M. Hsie, M.-Y. Wu, and C. Y. Huang, "Optimal urban sewer layout design using Steiner tree problems," *Engineering Optimization*, vol. 51, no. 11, pp. 1980–1996, 2019.

[10] F. K. Hwang and D. S. Richards, "Steiner tree problems," *Networks*, vol. 22, no. 1, pp. 55–89, 1992.

[11] M. S. Levin and R. I. Nuriakhmetov, "Multicriteria Steiner tree problem for communication network," *arXiv preprint arXiv:1102.2524*, 2011.

[12] I. Ljubić, "Solving Steiner trees: Recent advances, challenges, and perspectives," *Networks*, vol. 77, no. 2, pp. 177–204, 2021.

[13] M. Müller-Hannemann and S. Tazari, "A near linear time approximation scheme for Steiner tree among obstacles in the plane," *Computational Geometry*, vol. 43, no. 4, pp. 395–409, 2010.

[14] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org, 2017.

[15] V. Parque, "On hybrid heuristics for Steiner trees on the plane with obstacles," in *Evolutionary Computation in Combinatorial Optimization: 21st European Conference, EvoCOP 2021, Held as Part of EvoStar 2021, Virtual Event, April 7–9, 2021, Proceedings 21*. Springer, 2021, pp. 120–135.

[16] QGIS Development Team, *QGIS Geographic Information System*, QGIS Association, 2023. [Online]. Available: https://www.qgis.org

[17] S. B. Rao and W. D. Smith, "Approximating geometrical graphs via "spanners" and "banyans"," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, 1998, pp. 540–550.

[18] D. Rehfeldt and T. Koch, "Implications, conflicts, and reductions for Steiner trees," *Mathematical Programming*, vol. 197, pp. 903 – 966, 2023.

[19] M. Rosenberg, T. French, M. Reynolds, and L. While, "A genetic algorithm approach for the euclidean Steiner tree problem with soft obstacles," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2021, pp. 618–626.

[20] H. Tang, G. Liu, X. Chen, and N. Xiong, "A survey on steiner tree construction and global routing for vlsi design," *IEEE Access*, vol. 8, pp. 68 593–68 622, 2020.

[21] J. Y. Yen, "An algorithm for finding shortest routes from all source nodes to a given destination in general networks," *Quarterly of applied mathematics*, vol. 27, no. 4, pp. 526–530, 1970.

[22] M. Zachariasen and P. Winter, "Obstacle-avoiding Euclidean Steiner trees in the plane: an exact algorithm," in *Workshop on Algorithm Engineering and Experimentation*. Springer, 1999, pp. 286–299.

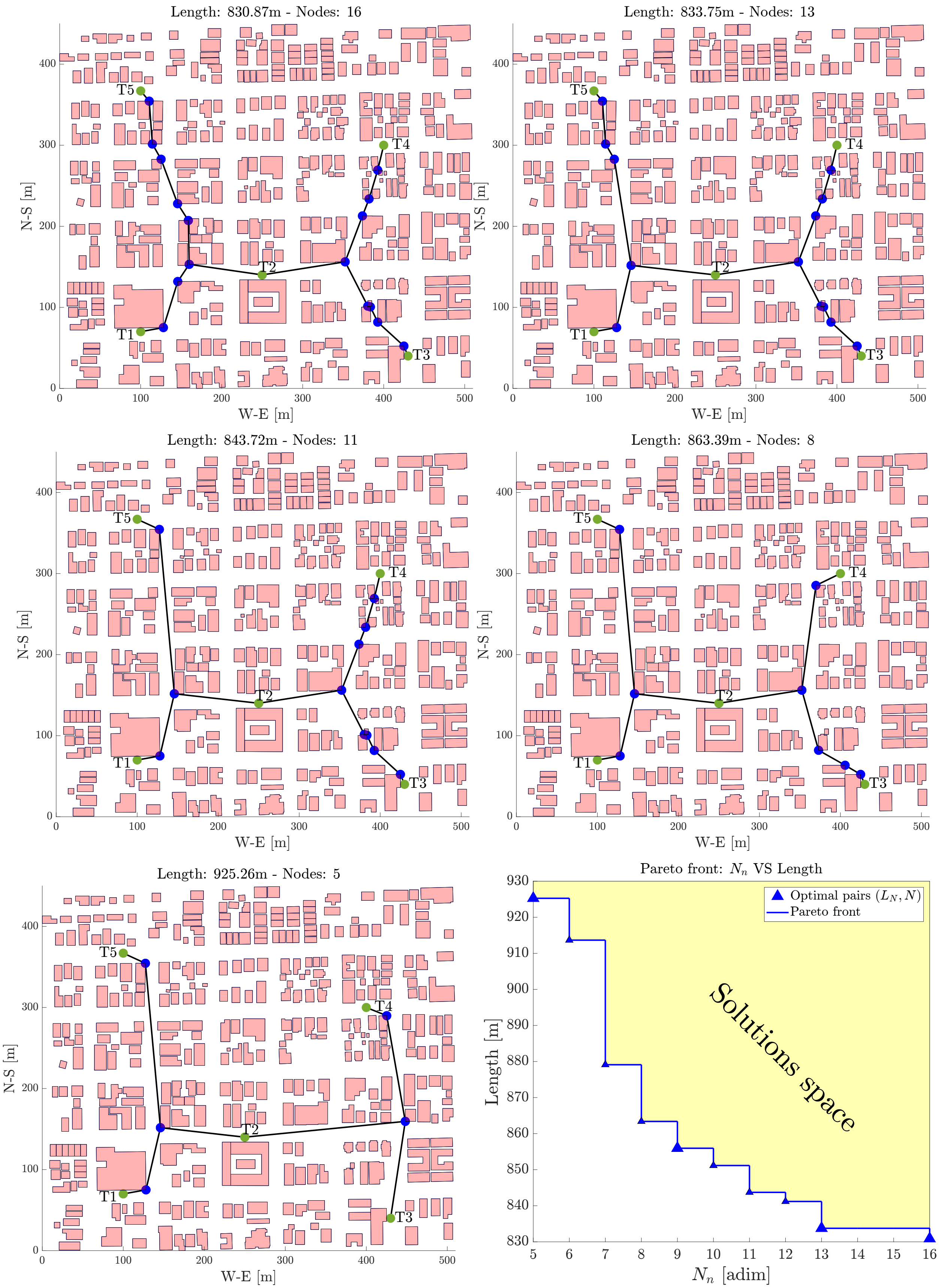

Fig. 6. Application of method LoSV-Yen to a setup with a large number of obstacles. The solution of the STPG yields a network with optimal length but with a number of nodes that needs not be the minimum. By means of branch alteration and n_{d3} displacement it is possible to lower the number of nodes, at the expense of the length of the network.