POLITECNICO DI TORINO
Repository ISTITUZIONALE

Robust Localization of UAVs in OTFS-Based Networks

Original
Robust Localization of UAVs in OTFS-Based Networks / Nordio, A., Chiasserini, C.F., Viterbo, E.. - ELETTRONICO. - (2023), pp. 7471-7477. (IEEE GLOBECOM 2023 Kuala Lumpur (Malaysia) 04-08 December 2023) [10.1109/GLOBECOM54140.2023.10437569].

Availability:
This version is available at: 11583/2980933 since: 2023-08-04T09:11:13Z

Publisher:
IEEE

Published
DOI:10.1109/GLOBECOM54140.2023.10437569

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2023 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

22 June 2026

Robust Localization of UAVs in OTFS-based Networks

Alessandro Nordio
CNR-IEIIT
Torino, Italy

Carla Fabiana Chiasserini
Politecnico di Torino
Torino, Italy

Emanuele Viterbo
Monash University
Melbourne, Australia

***Abstract*—We consider the problem of accurately localizing N unmanned aerial vehicles (UAV) in 3D space where the UAVs are part of a swarm and communicate with each other through OTFS modulated signals. Each receiving UAV estimates the multipath wireless channel on each link formed by the line-of-sight (LoS) transmission and by the single reflections from the remaining $N-2$ UAVs. The estimated power delay profiles are communicated to an edge server, which is in charge of computing the exact location of the UAVs. To obtain the UAVs locations, we propose an iterative algorithm, named Turbo Iterative Positioning (TIP), which, using the belief-propagation approach, effectively exploits the time difference of arrival (TDoA) measurements between the LoS and the non-LoS paths. Enabling a full cold start (no prior knowledge), our solution first maps each TDoA's profile element to a specific ID of the reflecting UAV's. The localization of the N UAVs is then derived via gradient descent optimization, with the aid of turbo-like iterations that can progressively correct some of the residual errors in the initial ID mapping operation. Our numerical results, obtained also using real-world traces, show how the multipath links are beneficial to achieving very accurate localization of all UAVs, even with a limited delay resolution.**

I. Introduction

Precise localization is a fundamental component of a wide range of applications that 5G-and-beyond communication networks are expected to enable [1]. Examples include safety applications in vehicular networks as well area exploration, rescue and relief work through unmanned aerial vehicles (UAV) swarms [2], [3]. In GPS-limited or denied environments, localization of such communicating nodes can be typically performed by measuring their distances, which, in turn, can be obtained by estimating the propagation time-delay of pilot signals. Nevertheless, and in spite of the large body of work existing on the use of orthogonal frequency-division multiplexing (OFDM) waveforms for this purpose [4], [5], precise localization in non-line-of-sight (nLoS) and high-mobility scenarios is still an open challenge [6]. To overcome this issue, orthogonal time-frequency space (OTFS) has been been recently considered as an alternative to OFDM. In particular, [7] shows the effectiveness of using OTFS-modulated physical random access channel (PRACH) transmissions for time-of-arrival based ranging, while [6] designs an OFTS transceiver for positioning-reference-signals transmission and reception, and uses it to implement a belief propagation-based cooperative positioning.

In this paper, we consider a UAV swarm where nodes communicate with each other as well as with an edge-server through OTFS-modulated signals. Unlike previous work, we envision precise localization of all the swarm UAVs without relying on positioning-specific signaling. Based on the time difference of arrival (TDoA) between line-of-sight (LoS) and the non-LoS paths, the UAVs can operate without requiring sample-scale synchronization (within $1/B$ [s] for a communications bandwidth B) among them. Specifically, our solution can deliver accurate positioning with only a loose frame synchronization (within M_d/B [s] for M_d delay bins in the delay-Doppler domain).

The low-resolution estimates of the delays between the multiple paths in the channels between all the pairs of UAV's are delivered to the edge server, which first maps such delay values onto the UAVs' identities. To do so, it exploits belief-propagation (BP), a method that, thanks to its low complexity and flexibility, has proved to work effectively for solving localization problems [8], [9]. Then the edge server uses such information to estimate the UAV's positions, by applying the gradient descend algorithm converging to the true values of the UAVs' positions.

The solution we envision, named turbo-iterative positioning (TIP), has limited complexity and does not require the availability of large bandwidth, as typically needed for precise localization. Furthermore, using also real-world UAV traces, we show that TIP provides highly accurate estimated positions and proves to be very robust to noise.

The rest of the paper is organized as follows. After detailing the assumptions and formulating the problem in Sec. II, we present our solution to identify the UAV's in the list of reflected paths identified by channel estimation in Sec. III. Then Sec. IV and Sec. V present our positioning algorithm, while Sec. VI shows some numerical results, also obtained using real-world UAV traces. Conclusions and future research directions are discussed in Sec. VII.

II. System Model and Problem Statement

We consider a swarm of N UAV's in a 3D region. The position of the i-th UAV is denoted by vector $\vec{P}_i$, whose components are expressed in arbitrary distance units and refer to a common Cartesian reference system. As depicted in

This work was partially supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 - program "RESTART").

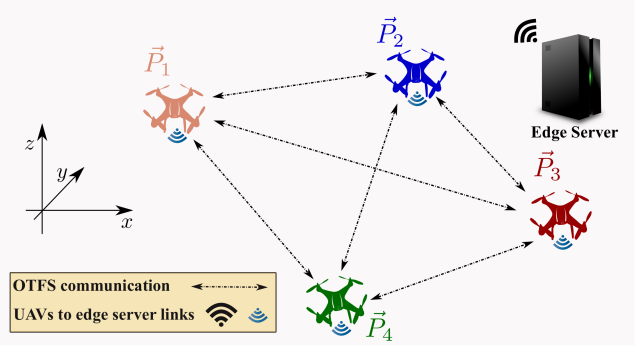

Fig. 1. Communicating UAVs assisted by an edge server.

Fig. 1, the UAVs communicate with each other using OTFS modulation, as well as with an edge server controlling the geographical area of interest. The wireless channel connecting any two UAVs is subject to multipath fading. For the sake of simplicity, we assume that no physical obstacle, other than the UAVs belonging to the swarm, can reflect or block the signal. Therefore, each communication channel between UAVs is characterized by the line-of-sight (LoS) path and $N-2$ nLoS components due to a single reflection.

In a practical scenario where the UAVs' transceivers are not synchronized with respect to a common time reference, they align onto the LoS path, which is associated to a zero delay. It follows that the delay-domain expression of the channel connecting UAVs i and $j \neq i$ is given by:

$$h_{i,j}(\tau) = c_{i,j}\delta(\tau) + \sum_{k=1,k\neq\{i,j\}}^{N} c_{i,j,k}\delta(\tau - \tau_{i,j,k}) \quad (1)$$

where $c_{i,j}$ and $c_{i,j,k}$ are, respectively, the LoS and k-th nLoS channel coefficient, whereas $\tau_{i,j,k}$ corresponds to the TDoA between the k-th non-LoS path and the LoS path. For simplicity, we omit the time dependence in the channel coefficients due to mobility, since by using OTFS we can effectively decouple the delays from the Doppler shifts in the delay-Doppler (DD) domain [10].

Hence $\tau_{i,j,k} = \tau_{i,k} + \tau_{k,j} - \tau_{i,j}$, where $\tau_{u,v} = |\vec{P}_u - \vec{P}_v|/c$, $u, v \in \{i, j, k\}$, $|\cdot|$ denotes the norm of vector, and c is the speed of light. Delays $\tau_{i,j,k}$ clearly depend upon the swarm geometry and for each of them, we define the corresponding distance

$$\delta_{i,j,k} \triangleq c\,\tau_{i,j,k} = |\vec{P}_j - \vec{P}_k| + |\vec{P}_k - \vec{P}_i| - |\vec{P}_j - \vec{P}_i|\,. \quad (2)$$

Notice also that, due to channel reciprocity, $\delta_{i,j,k} = \delta_{j,i,k}$ for all i, j, and $k \neq \{i, j\}$. The delays in (1) or, equivalently, the distances in (2) are typically measured through pilot-based channel estimation, to some degree of accuracy dependent on the bandwidth [11].

In OTFS, the DD domain is discretized into an $M_d \times N_D$ grid resulting in a delay resolution of $\Delta\tau = T/M_d = 1/B$ and a Doppler shift resolution of $\Delta\nu = 1/(N_D T)$, where $T = 1/\Delta f$ is the duration of a block in the frame of duration $T_f = N_D T$ and $B = M_d \Delta f$ is the communication channel bandwidth. Simple channel estimation techniques, such as in [11], enable locating the delay and Doppler shift of each path with the above resolution. By partitioning the DD domain grid into disjoint rectangular tiles, UAVs can simultaneously broadcast a pilot in a single frame, thus allowing all the receivers to perform channel estimation simultaneously within T_f seconds [10]. In turn, this allows estimating all the TDoA's $\tau_{i,j,k}$ and the corresponding distance differences

$$\widetilde{\delta}_{i,j,k} = \delta_{i,j,k} + \eta_{i,j,k} \quad (3)$$

where $\eta_{i,j,k}$ is a random variable with density $f_\eta(\cdot)$, representing the estimation error due to the limited resolution $\Delta\tau$ and noise. While the noise component of $\eta_{i,j,k}$ can be reduced by increasing the pilot power, the quantization noise can only be reduced by increasing the bandwidth resource.

We assume that the j-to-i channel delay profile estimations are available at each receiver $i \neq j$ as an ordered list

$$\Lambda_{i,j} = \{d_{i,j,m} | m = 1, \ldots, N-2\} \quad (4)$$

with elements arranged in increasing order. We also assume that lists $\Lambda_{i,j}$'s are sent by the UAVs to the edge server controlling the geographical area. Then we have $d_{i,j,m} = \widetilde{\delta}_{i,j,k}$, if the m-th path in the distance associated with the ordered list $\Lambda_{i,j}$ corresponds to the k-th UAV. In other words, we can associate with each list $\Lambda_{i,j}$ a bijective map

$$\mu_{i,j} : \{1, \ldots, N-2\} \to \{1, \ldots, N\} \setminus \{i, j\}$$

such that $\mu_{i,j}(m) = k$. It is clear that, if the UAVs' positions are unknown, then the receiver cannot associate elements $d_{i,j,m}$'s with the UAVs that generated them, i.e., it has no knowledge of maps $\mu_{i,j}$'s. Then the problem we address in this work can be stated as follows:

Problem – *Given N UAVs communicating using OTFS through a channel as in* (1) *and the lists of noisy delay profile estimations $\Lambda_{i,j}$, how can an edge server reliably estimate the UAVs' positions, $\vec{P}_i$, $i = 1 \ldots N$?*

We tackle this problem in two steps:

1) We estimate maps $\mu_{i,j}$, given the sets $\Lambda_{i,j}$, $\forall i, j = 1, \ldots, N$ and $i \neq j$. Towards this goal, we will relax the deterministic maps $\mu_{i,j}$ to probabilistic maps $\mathcal{M}_{i,j}$, and use a BP approach to obtain an estimation of the former, denoted by $\widehat{\mu}_{i,j}$, from the latter (Sec. III).
2) We estimate UAVs' positions, given the sets $\Lambda_{i,j}$ and the estimates $\widehat{\mu}_{i,j}$ (Sec. IV).

The two steps are then combined in a high-performance turbo algorithm, named Turbo Iterative Positioning (TIP), for providing highly reliable UAVs' position estimates $\vec{P}_i^\star$ (Sec. V).

III. Mapping Delay Profiles to UAVs' Identities

To estimate maps $\mu_{i,j}$'s, let us first focus on a subset of four UAVs, respectively labelled by i, j, k, h, as depicted in Fig. 2. The figure also highlights the true distances between the UAVs, $|\vec{P}_u - \vec{P}_v|$, for $u, v \in \{i, j, k, h\}$. By recalling the

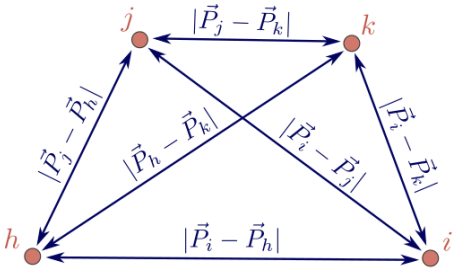

Fig. 2. A subset of four UAVs and their true distances.

expression of $\delta_{i,j,k}$ in (2), it is easy to observe that in the absence of noise, for all quadruples of UAVs in the swarm

$$\delta_{i,j,k} - \delta_{i,j,h} + \delta_{i,k,h} - \delta_{j,h,k} = 0 \tag{5}$$

where $i,j,k,h = 1,\dots,N$, $j \neq i$, $k \neq \{i,j\}$, $h \neq \{i,j,k\}$. This implies that one can find the correct associations $\mu_{i,j}$ between the entries of the list $\Lambda_{i,j}$ and the UAVs' identities by searching the elements of lists $\Lambda_{i,j}$, $\Lambda_{i,k}$, and $\Lambda_{j,h}$ until a relation such as the one in (5) holds.

More precisely, given the m-th element $d_{u,v,m}$ of the ordered list $\Lambda_{u,v}$, if for some integers m,n,s,t the relation

$$d_{i,j,m} - d_{i,j,n} + d_{i,k,s} - d_{j,h,t} = 0 \tag{6}$$

is satisfied, then, by comparing (5) to (6), we deduce the maps $\mu_{i,j}(m) = k$, $\mu_{i,j}(n) = h$, $\mu_{i,k}(s) = h$, and $\mu_{j,h}(t) = k$. Notice that there are $O(N^8)$ constraint equations like (6) that need to be satisfied simultaneously by the d's in all the lists Λ. A brute force search would have an exponential complexity of $O([(N-2)!]^{N(N-1)})$. Additionally, whenever the distance estimations are affected by noise, the l.h.s. of (6) is, in general, not 0. To find the maps in the presence of noise in an efficient way, we relax the deterministic maps $\mu_{i,j}$'s to probabilistic maps $\mathcal{M}_{i,j}$ whose m-th element, $\mathcal{M}_{i,j}(m)$, has (pmf) $\pi_{i,j,m,k} \triangleq \mathbb{P}(\mathcal{M}_{i,j}(m) = k)$ for $k \in \{1,\dots,N\}/\{i,j\}$. In the absence of noise, the pmf of $\mathcal{M}_{i,j}(m)$ coincides with a distribution taking on 1 in correspondence of the true value $\mu_{i,j}(m)$.

Let us define the set of all the maps as $\boldsymbol{\mu} = \{\mu_{u,v}, u \neq v\}$ and the set of all the random maps as $\boldsymbol{\mathcal{M}} = \{\mathcal{M}_{u,v}, u \neq v\}$. In order to derive accurate estimates $\widehat{\mu}_{i,j}$ of the maps $\mu_{i,j}$, we should solve the problem

$$\widehat{\boldsymbol{\mu}} = \arg\max_{\boldsymbol{\mu}} \mathbb{P}\left(\boldsymbol{\mathcal{M}} = \boldsymbol{\mu}\right) \tag{7}$$

which has a complexity $O([(N-2)!]^{N(N-1)})$ and it is in any case intractable since the joint pmf of $\boldsymbol{\mathcal{M}}$ is unknown. To overcome this problem, we propose the heuristic greedy approach described in Algorithm 1 which relies on the marginals $\pi_{i,j,m,k}$ and has polynomial complexity $O\left(N(N-1)(N-2)\right)$.

Assuming that all $\pi_{i,j,m,k}$ are known, for each map $\mu_{i,j}$, we collect them in the $(N-2)\times N$ matrices $\boldsymbol{\Pi}$, for each i,j. We then work iteratively and, at each step, we find the most likely association in the map. Specifically, we seek the largest entry of $\boldsymbol{\Pi}$, we record its row index m' and its column index k', and we set $\widehat{\mu}_{i,j}(m') = k'$. Next, we set to zero the m'-th row and the k'-th column of $\boldsymbol{\Pi}$: this operation is necessary because no other element of the map can be assigned to value k' in the following steps of the algorithm. The procedure ends when all values of the elements of $\boldsymbol{\Pi}$ become zero, i.e., a decision is made on all the elements of the map $\widehat{\mu}_{i,j}$.

In order to compute the marginals $\pi_{i,j,m,k}$, we apply the BP approach, as described in the following. In the BP terminology, the random maps $\mathcal{M}_{i,j}$ represent the *variable nodes*, whereas the *check nodes* are defined by the equations

$$d_{i,j,\mathcal{M}_{i,j}^{-1}(k)} - d_{i,j,\mathcal{M}_{i,j}^{-1}(h)} + d_{i,k,\mathcal{M}_{i,k}^{-1}(h)} - d_{j,h,\mathcal{M}_{j,h}^{-1}(k)} \tag{8}$$

which specify the variable nodes involved in the check node. Notice that the check nodes coincide with the l.h.s. of (6), involving the variable nodes in the subscripts. In the BP model, the marginals, $\pi_{i,j,m,k}$, represent the messages flowing from the variable nodes, $\mathcal{M}_{i,j}$, to the check nodes.

The messages from check nodes to variable nodes are defined below. Let $Z^{i,j,m}$ be the random variable

$$Z^{i,j,m} = d_{i,j,m} - X_1 + X_2 - X_3 \tag{9}$$

where X_1, X_2, X_3 are discrete random variables taking values $x_1 \in \Lambda_{i,j} \setminus \{d_{i,j,m}\}$, $x_2 \in \cup_{u \neq j} \Lambda_{i,u}$, and $x_3 \in \cup_{u \neq \{i,v(x_2)\}} \Lambda_{j,u}$, respectively, where x_2 is selected from $\Lambda_{i,v(x_2)}$. It is clear that such random variables are correlated since the choice of X_1 influences the set of possible values of X_2 and similarly for X_2 and X_3. Next, define the events

$$\begin{aligned} A^{i,j}_{k,h,n,s,t} &= \{\mathcal{M}_{i,j}(n)=h, \mathcal{M}_{i,k}(s)=h, \mathcal{M}_{j,h}(t)=k\}, \\ B^{i,j,m}_{k,h,n,s,t} &= \{\mathcal{M}_{i,j}(m)=k, A^{i,j}_{k,h,n,s,t}\}. \end{aligned}$$

Then, the density of $Z^{i,j,m}$ is given by:

$$\begin{aligned} f(z^{i,j,m}) &= \sum_{\substack{k \neq \{i,j\}, h \neq k \\ n,s,t}} f(z^{i,j,m} | B^{i,j,m}_{k,h,n,s,t}) \mathbb{P}\left(B^{i,j,m}_{k,h,n,s,t}\right) \\ &= \sum_{\substack{k \neq \{i,j\}, h \neq k \\ n,s,t}} g\left(z^{i,j,m}_{k,h,n,s,t}\right) \mathbb{P}\left(B^{i,j,m}_{k,h,n,s,t}\right) \end{aligned} \tag{10}$$

where $z^{i,j,m}_{k,h,n,s,t}$ is the l.h.s. of (6) and $g(\cdot)$ a distribution accounting for the sum of 4 distance measurement noise terms, whose density is $g(\cdot) = f_\eta(\cdot) * f_\eta(\cdot) * f_\eta(\cdot) * f_\eta(\cdot)$, with $*$ denoting the convolution operator.

The messages from the check nodes to the variable nodes are the conditional probabilities

$$\begin{aligned} p_{i,j,m,k} &\triangleq \mathbb{P}(\mathcal{M}_{i,j}(m) = k | z^{i,j,m}) \\ &= \frac{f(z^{i,j,m} | \mathcal{M}_{i,j}(m) = k) \cdot \pi_{i,j,m,k}}{f(z^{i,j,m})} \\ &\approx \frac{\pi_{i,j,m,k}}{f(z^{i,j,m})} \sum_{\substack{h \neq \{i,j,k\} \\ n,s,t}} g\left(z^{i,j,m}_{k,h,n,s,t}\right) \cdot \\ &\qquad \cdot \pi_{i,j,n,h} \cdot \pi_{i,k,s,h} \cdot \pi_{j,h,t,k} \end{aligned} \tag{11}$$

where

$$f\left(z^{i,j,m}|\mathcal{M}_{i,j}(m)=k\right) = \sum_{\substack{h\neq\{i,j,k\},\\ n,s,t}} g\left(z^{i,j,m}_{k,h,n,s,t}\right)\mathbb{P}\left(A^{i,j}_{h,n,s,t}\right)$$

and, in the last line of (11), we assumed independence among the events $\mathcal{M}_{i,j}(n)=h$, $\mathcal{M}_{i,k}(s)=h$, and $\mathcal{M}_{j,h}(t)=k$, so that $\mathbb{P}(A^{i,j}_{h,n,s,t}) \approx \pi_{i,j,n,h}\cdot\pi_{i,k,s,h}\cdot\pi_{j,h,t,k}$.

The iterative procedure that leads to the computation of the probabilities $\pi_{i,j,m,k}$ is described in Algorithm 2. For the ℓ-th iteration, based on (11) we compute:

$$p^{(\ell)}_{i,j,m,k} \leftarrow \sum_{\substack{h\neq\{i,j,k\}\\ n,s,t}} g\left(z^{i,j,m}_{k,h,n,s,t}\right)\cdot\pi^{(\ell-1)}_{i,j,n,h}\cdot\pi^{(\ell-1)}_{i,k,s,h}\cdot\pi^{(\ell-1)}_{j,h,t,k}\,. \quad (12)$$

The marginals are then updated as

$$\pi^{(\ell)}_{i,j,m,k} \leftarrow \frac{p^{(\ell)}_{i,j,m,k}}{\sum_{u\neq\{i,j\}} p^{(\ell)}_{i,j,m,u}} \quad (13)$$

where $\pi^{(0)}_{i,j,m,k} \leftarrow \frac{1}{N-2}$. Note that in (12) we removed the term $\pi^{(\ell-1)}_{i,j,m,k}$ appearing in (11) since it provides highly correlated information with $p^{(\ell)}_{i,j,m,k}$. Hence, the algorithm only employs the *extrinsic information* provided by the other marginals.

Once the BP algorithm reaches convergence or a sufficient number of iterations has been performed, the resulting marginal pmf's, $\pi_{i,j,m,k}$ are fed to Algorithm 1.

Algorithm 1 $\widehat{\boldsymbol{\mu}} = \text{EstimateMaps}(\boldsymbol{\Lambda})$

Require: $N \geq 2$;
  $\{\pi_{i,j,m,k}\} \leftarrow \text{ComputeMarginals}(\boldsymbol{\Lambda})$;
  for $i,j = 1,\ldots,N$, $i \neq j$ **do**
    $[\boldsymbol{\Pi}]_{m,k} \leftarrow \begin{cases} \pi_{i,j,m,k} & k \neq \{i,j\} \\ 0 & k = i \text{ or } k = j \end{cases}$
    while $\boldsymbol{\Pi} \neq \mathbf{0}$ **do**
      $[m',k'] \leftarrow \arg\max_{m,k}([\boldsymbol{\Pi}]_{m,k})$;
      $\widehat{\mu}_{i,j}(m') \leftarrow k'$;
      $[\boldsymbol{\Pi}]_{m',q} \leftarrow 0$ for $q = 1,\ldots,N$;
      $[\boldsymbol{\Pi}]_{q,k'} \leftarrow 0$ for $q = 1,\ldots,N-2$;
  return $\widehat{\boldsymbol{\mu}}$

Algorithm 2 $\{\pi_{i,j,m,k}\} = \text{ComputeMarginals}(\boldsymbol{\Lambda})$

Require: $N \geq 2$, $\boldsymbol{\Lambda}$, $I_\mu > 0$;
  $\pi^{(0)}_{i,j,m,k} \leftarrow \frac{1}{N-2}$, for every $i\neq j$, $m=1,\ldots,N-2$, $k\neq\{i,j\}$
  for $\ell = 1,\ldots,I_\mu$ **do**
    for $i,j = 1,\ldots,N$, $j \neq i$, $m = 1,\ldots,N-2$ **do**
      for $k = 1,\ldots,N$, $k \neq \{i,j\}$ **do**
        compute $p^{(\ell)}_{i,j,m,k}$ using (12);
      for $k = 1,\ldots,N$, $k \neq \{i,j\}$ **do**
        compute $\pi^{(\ell)}_{i,j,m,k}$ using (13);
  return $\{\pi^{(I_\mu)}_{i,j,m,k}\}$

IV. Estimating the Nodes' Positions

Given the maps $\mu_{i,j}$'s, the edge server can use them to estimate the UAV's positions. The approach we propose proceeds iteratively, computing the square error between the channel estimates and the tentative normalized distances and applying a gradient descent algorithm to minimize such an objective function.

Let $\vec{\mathbf{Q}} \triangleq \{\vec{Q}_1,\ldots,\vec{Q}_N\}$ be a set of tentative decisions for the positions of the N UAVs. Then, an estimate $\vec{P}^\star_i$ of the positions $\vec{P}_i$ can be obtained as

$$\vec{\mathbf{P}}^\star = \arg\min_{\vec{\mathbf{Q}}} \mathcal{E}(\vec{\mathbf{Q}}) \quad (14)$$

where $\vec{\mathbf{P}}^\star \triangleq \{\vec{P}^\star_1,\ldots,\vec{P}^\star_N\}$ and

$$\mathcal{E}(\vec{\mathbf{Q}}) \triangleq \sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}\sum_{\substack{k=1\\ k\neq i,j}}^{N}\left(\widehat{\delta}_{i,j,k} - q_{i,j,k}\right)^2 \quad (15)$$

is the square error between the channel measurements, $\widehat{\delta}_{i,j,k}$, and the tentative normalized distances $q_{i,j,k}$, which, as in (2), are defined as $q_{i,j,k} \triangleq |\vec{Q}_j - \vec{Q}_k| + |\vec{Q}_k - \vec{Q}_i| - |\vec{Q}_j - \vec{Q}_i|$.

The square error in (15) is, in general, a non-convex function, however local or global minima can be easily found by applying a standard gradient descent method.

Let $\vec{\mathbf{Q}}^{(\ell)}$ be an estimate of the position vectors at iteration ℓ. Then, by applying one gradient descent step, the estimate at iteration $\ell+1$ is given by $\vec{\mathbf{Q}}^{(\ell+1)} = \vec{\mathbf{Q}}^{(\ell)} - \gamma^{(\ell)}\vec{\mathbf{G}}^{(\ell)}$ where $\gamma^{(\ell)}$ denotes the step size and $\vec{\mathbf{G}}^{(\ell)} = [\vec{G}^{(\ell)}_1,\ldots,\vec{G}^{(\ell)}_N]$ is the gradient. After some algebra, we get

$$\vec{G}_h(\vec{\mathbf{Q}}) = \frac{\partial\mathcal{E}(\vec{\mathbf{Q}})}{\partial\vec{Q}_h} = 2\sum_{\substack{i=1\\ i\neq h}}^{N}\sum_{\substack{j=1\\ j\neq h,i}}^{N}\begin{array}{r}(w_{h,i,j}+w_{i,h,j})\,(\vec{u}_{h,i}-\vec{u}_{h,j})\\ -w_{i,j,h}\,(\vec{u}_{h,j}+\vec{u}_{h,i})\end{array}$$

where $w_{i,j,k} \triangleq \widehat{\delta}_{i,j,k} - q_{i,j,k}$ and $\vec{u}_{i,j} = -\vec{u}_{j,i} = \frac{\vec{Q}_i-\vec{Q}_j}{|\vec{Q}_i-\vec{Q}_j|}$ is the versor pointing from j to i.

The gradient descent algorithm starts from an initial estimate $\vec{\mathbf{Q}}^{(0)}$ of the nodes' positions and iterates until the following stopping condition on the square error

$$\frac{\mathcal{E}(\vec{\mathbf{Q}}^{(\ell+1)}) - \mathcal{E}(\vec{\mathbf{Q}}^{(\ell)})}{\mathcal{E}(\vec{\mathbf{Q}}^{(\ell)})} < \epsilon$$

is met, or a maximum number of iterations is reached.

V. Turbo Iterative Positioning

We now combine the methods presented in Sections III and IV to design our TIP, a high-performance iterative algorithm for estimating the positions of the UAVs. The block diagram of TIP is depicted in Fig. 3, and the corresponding pseudocode is outlined in Algorithm 3.

We recall that the UAVs communicate with each other and obtain estimates of the paths delays or, equivalently, the set of lists $\boldsymbol{\Lambda} = \{\Lambda_{i,j}\}$, which are sent to the edge server running the TIP algorithm. As explained in Sec. II, the elements of list $\Lambda_{i,j}$ are the estimated distances $d_{i,j,m}$, $m = 1,\ldots,N-2$, ordered in increasing order.

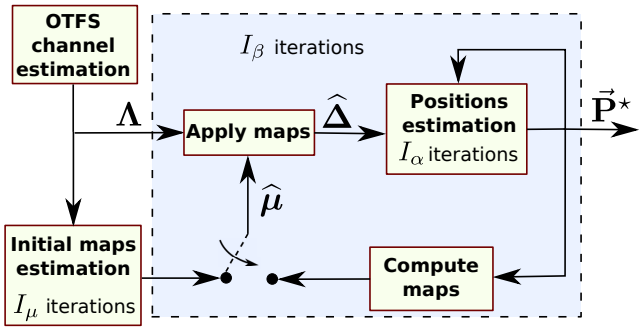

Fig. 3. Scheme of the TIP algorithmic framework.

The TIP first derives the estimated maps $\widehat{\boldsymbol{\mu}}$ from the set of lists $\boldsymbol{\Lambda}$, using Algorithm 1. Such estimated maps are then applied to lists $\boldsymbol{\Lambda}$. In particular, the estimated map $\widehat{\mu}_{i,j}$ is applied to list $\Lambda_{i,j}$, so as to reorder its elements according to the UAVs' identity index. So doing, we obtain list $\widehat{\Delta}_{i,j}$ where the k-th element, $\widehat{\delta}_{i,j,k}$, is an estimate of the distance $\delta_{i,j,k}$, defined in (2), due to the reflection by the k-th UAV.

The set of reordered lists $\widehat{\boldsymbol{\Delta}} = \{\widehat{\Delta}_{i,j}\}$ are then fed to the gradient descend algorithm described in Sec. IV. The estimated positions, $\vec{\mathbf{P}}^\star$, are the output of the TIP algorithm. They can however be fed back to the algorithm in an iterative fashion, to improve the accuracy of the position estimates. Indeed, from $\vec{\mathbf{P}}^\star$ it is possible to compute better estimates of maps $\boldsymbol{\mu}$ which, in turn, can be used to obtain a more reliable reordering of the lists in $\boldsymbol{\Lambda}$.

In summary, from positions $\vec{\mathbf{P}}^\star$ and using (2), we first compute the list of distances

$$\Delta^\star_{i,j} = \{\delta^\star_{i,j,k} | k \in \{1,\dots,N\} \setminus \{i,j\}\}$$

with the k-th element

$$\delta^\star_{i,j,k} \quad = \quad |\vec{P}^\star_j - \vec{P}^\star_k| + |\vec{P}^\star_k - \vec{P}^\star_i| - |\vec{P}^\star_j - \vec{P}^\star_i| \,. \quad (16)$$

The elements of $\Delta^\star_{i,j}$ are then ordered in ascending order to form lists $\Lambda^\star_{i,j}$. Finally, each $\widehat{\mu}_{i,j}$ is updated as the map that transforms $\Lambda^\star_{i,j}$ into $\Delta^\star_{i,j}$.

Algorithm 3 TIP algorithm

Require: $N \geq 2$, $\boldsymbol{\Lambda}$, I_β
  $\widehat{\boldsymbol{\mu}} \leftarrow$EstimateMaps($\boldsymbol{\Lambda}$);
  $\vec{\mathbf{P}}^\star \leftarrow$ rand;
  for $\ell \leftarrow 0$ to I_β **do**
    Apply the maps $\widehat{\boldsymbol{\mu}}$ to $\boldsymbol{\Lambda}$ to obtain $\widehat{\boldsymbol{\Delta}}$;
    $\{\vec{\mathbf{P}}^\star, \mathcal{E}\} \leftarrow$GradientDescent($\widehat{\boldsymbol{\Delta}}, \vec{\mathbf{P}}^\star$);
    Compute the set of lists $\boldsymbol{\Delta}^\star$ using $\vec{\mathbf{P}}^\star$ and (16);
    Sort each list in $\boldsymbol{\Delta}^\star$ in ascending order to obtain $\boldsymbol{\Lambda}^\star$;
    Compute the maps $\widehat{\boldsymbol{\mu}}$ that yield $\boldsymbol{\Delta}^\star$ from $\boldsymbol{\Lambda}^\star$;
    $\ell \leftarrow \ell + 1$;
  return $\{\vec{\mathbf{P}}^\star, \mathcal{E}\}$;

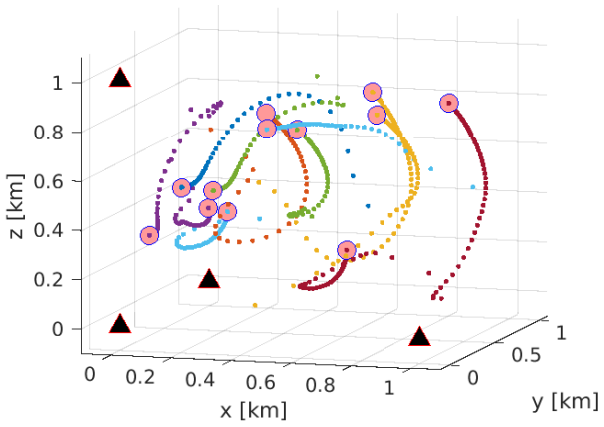

Fig. 4. Trajectories of the gradient descent localization. The circles are the exact locations and the four triangles are anchor nodes of known locations to find absolute positions.

VI. Results

To test the performance of the TIP algorithm, we consider a scenario where a swarm composed of N UAVs is located in a cubic region of side 1,000 m. To avoid translation and rotation ambiguities in the estimated positions, we assume that $A = 4$ UAVs (out of N) have a known position, i.e., they serve as *anchors* while estimating the positions of the remaining $N - A$ UAVs. Fig. 4 shows an example of the trajectories of the gradient descend algorithm, for the case $N = 16$ and $A = 4$ in the absence of measurement noise and for a perfect knowledge of the maps $\boldsymbol{\mu}$. The $A = 4$ triangles represent the anchors whereas the circles represent the exact location of the remaining $N - A$ UAVs. The sequence of points represent the estimates $\vec{\mathbf{P}}^\star$ as the gradient descent iteration index, I_α, varies. As can be observed, the sequence of position estimates rapidly converges to the exact locations of the UAVs.

As performance metric for TIP, we consider the root mean square error (RMSE) on the position estimate per UAV and per dimension. The RMSE is computed by averaging over several realizations of the true UAVs positions and is defined as

$$\text{RMSE} = \sqrt{\frac{1}{3(N-A)} \sum_{i=1}^{N-A} \mathbb{E}\left|\vec{P}^\star_i - \vec{P}_i\right|^2}$$

where the $\mathbb{E}[\cdot]$ is the average operator. In the following results, we consider a swarm composed of $N = 8$ UAVs, out of which $A = 4$ are anchors. In the gradient descent algorithm, we set the step $\gamma = 8 \cdot 10^{-3}$ and the stopping threshold $\epsilon = 10^{-4}$. In our simulations, the measurement noise terms ($\eta_{i,j,k}$ in (3)) are independent and uniformly distributed random variables defined in the range $[-c\Delta\tau/2, c\Delta\tau/2]$. Fig. 5(left) shows the RMSE achieved by TIP versus the number of iterations of the gradient descend algorithm, I_α, for $I_\beta = 0$ (no TIP iterations) and a signal bandwidth $B = 15$ MHz, corresponding to a discretization step $c\Delta\tau = 20$ m. The RMSE was obtained by averaging over 100 random realizations of the true UAVs

positions, uniformly distributed in a cube of side 1,000 m. One can notice that $I_\mu = 10$ iterations of the BP algorithm provide the same performance as a genie-aided (GA) algorithm, which has perfect knowledge of the maps $\boldsymbol{\mu}$. The GA curve is used here as benchmark since it represents a lower bound for the TIP RMSE. Fig. 5(left) shows that, despite 20 m discretization step in the measurements, the system achieves an RMSE of about 1 m after $I_\alpha = 50$ iterations, which means that BP provides very reliable estimates $\widehat{\boldsymbol{\mu}}$. However, for larger discretization steps ($B = 3$ MHz), such estimates have much lower reliability: some TIP iterations are then required to improve the positioning RMSE, as in the example shown in Fig. 5(right). Here the discretization step is set to 100 m and the number of BP iterations is $I_\mu = 10$. For $I_\beta = 0$ (no TIP iterations), the resulting RMSE is quite large, about 50 m, whereas a single TIP iteration lowers it at about 10 m. With a further TIP iteration ($I_\beta = 2$) the system reaches the GA performance.

Fig. 6(left) shows the RMSE plotted versus the signal bandwidth, B, as the number of TIP iterations, I_β, varies, for $I_\mu = 1$. In this case, the RMSE has been computed by averaging over the set of real UAV positions available in [12] whose coordinates have been linearly scaled so as to fit in our simulation scenario, i.e., a cube of side 1,000 m. We observe that, as B increases, the discretization step decreases and the system provides, in general, more accurate positioning. For $I_\beta = 5$, the system performance reaches the GA for all considered values of B in the range 3–300 MHz. For $I_\mu = 2$, as shown in Fig. 6(right), the convergence of the TIP performance to GA is even faster. Indeed, close to optimal performance is already achieved after only $I_\beta = 2$ TIP iterations.

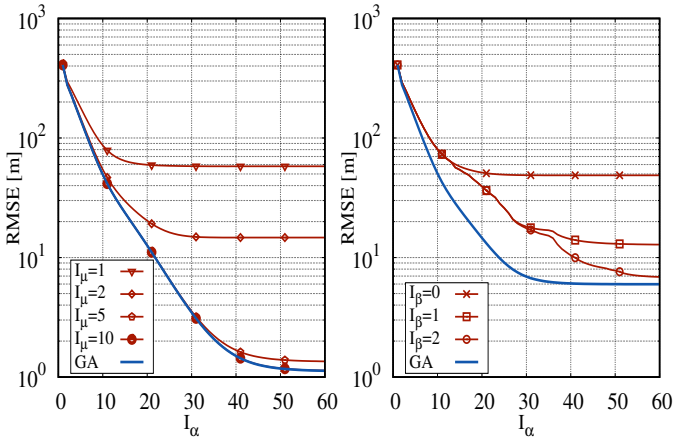

Fig. 5. RMSE as a function of the number of gradient descent iterations (I_α). Left: $I_\beta = 0$ and $B = 15$ MHz corresponding to $c\Delta\tau = 20$ m. Right: $I_\beta = 10$ and $B = 3$ MHz corresponding to $c\Delta\tau = 100$ m.

VII. Conclusions

We proposed an iterative algorithm for localizing UAVs communicating with each other, by using TDoA measurements

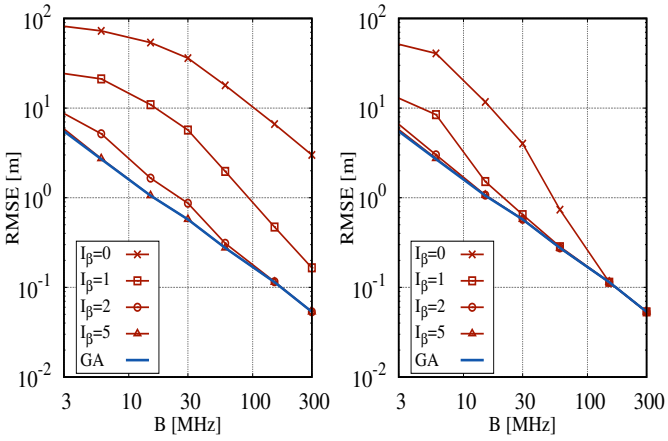

Fig. 6. RMSE as a function of the signal bandwidth as I_β varies, for $I_\mu = 1$ (left) and $I_\mu = 2$ (right).

obtained through OTFS-modulated signals. The algorithm exploits belief propagation and gradient descent optimization, to achieve precise localization even in the presence of severe estimation errors due to noise and a very limited resolution. The remarkable advantage of the proposed algorithm is that it enables to increase the localization accuracy of systems with limited bandwidth.

Future work will tackle the problem of UAV tracking, by estimating the nodes' speed beside their positions. We will also extend our method to include passive reflectors for terrestrial applications.

References

[1] K. Witrisal, P. Meissner, E. Leitinger, Y. Shen, C. Gustafson, F. Tufvesson, K. Haneda, D. Dardari, A. F. Molisch, A. Conti, and M. Z. Win, "High-accuracy localization for assisted living: 5G systems will turn multipath channels from foe to friend," *IEEE Signal Processing Magazine*, vol. 33, no. 2, pp. 59–70, 2016.

[2] A. R. Ansari, N. Saeed, M. I. Ul Haq, and S. Cho, "Accurate 3D localization method for public safety applications in vehicular ad-hoc networks," *IEEE Access*, vol. 6, pp. 20 756–20 763, 2018.

[3] S. M. Asaad and H. S. Maghdid, "A comprehensive review of indoor/outdoor localization solutions in IoT era: Research challenges and future perspectives," *Computer Networks*, vol. 212, p. 109041, 2022.

[4] L. Han, R. Liu, X. Lv, Z. Wang, and Q. Zhu, "Optimal waveform design for integrated localization and communication in OFDM systems," in *2022 IEEE 8th International Conference on Computer and Communications (ICCC)*, 2022, pp. 868–874.

[5] J. Bai, G. Wei, S. Wang, X. Wang, and Z. Fei, "Efficient direct localization of OFDM emitters in multipath environment with mobile receivers," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 1, pp. 545–556, 2022.

[6] M. J. Emadi, S. Hu, and H. Wang, "Precise positioning: When OTFS meets distributed cooperative positioning," Feb. 2023. [Online]. Available: 10.36227/techrxiv.21951014.v1

[7] F. Linsalata, A. Albanese, V. Sciancalepore, F. Roveda, M. Magarini, and X. Costa-Perez, "OTFS-superimposed PRACH-aided localization for UAV safety applications," in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 1–6.

[8] L. Wielandner, E. Leitinger, F. Meyer, and K. Witrisal, "Message passing-based 9-D cooperative localization and navigation with embedded particle flow," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 9, pp. 95–109, 2023.

[9] E. Leitinger, F. Meyer, F. Hlawatsch, K. Witrisal, F. Tufvesson, and M. Z. Win, "A belief propagation algorithm for multipath-based SLAM," *IEEE Transactions on Wireless Communications*, vol. 18, no. 12, pp. 5613–5629, 2019.

[10] Y. Hong, T. Thaj, and E. Viterbo, *Delay-Doppler Communications: Principles and Applications*. Academic Press, Elsevier, 2022.

[11] P. Raviteja, K. T. Phan, and Y. Hong, "Embedded Pilot-Aided Channel Estimation for OTFS in Delay–Doppler Channels," *IEEE Trans. Veh. Tech.*, vol. 68, no. 5, pp. 4906–4917, 2019.

[12] "Drone tracking dataset," https://github.com/CenekAlbl/drone-tracking-datasets.