DIeSEL: DIstributed SElf-Localization of a network of vehicles

Pusheng Ji
pusheng.ji@tecnico.ulisboa.pt
Instituto Superior Técnico
Portugal

***Abstract*—Localization is a crucial component in navigation and proper data georeferencing. Absolute localization is nowadays possible using systems such as GPS, but economical or physical constraints may prevent their usage, for example, in GPS-denied indoor or underwater environments. In these cases, other schemes must be developed in order to successfully locate the nodes, such as integrating distance or angle information between the nodes to assess their relative position.**

The localization problem in a network of agents is a non-convex optimization problem, and, thus, very difficult to solve. Firstly, we derive DIeSEL, a range-based, distributed, parameter-free, and provably convergent localization method to localize individual devices or networks of autonomous agents. Although DIeSEL is a general formulation for any GPS-denied environment, with a few landmarks, range measurements and velocity information, we motivated most of our simulations to assessing DIeSEL's performance with the positioning of underwater vehicles. We present in this extended abstract the main set of results.

In addition to the range-based strategy, we extend DIeSEL into DIeSEL-B, to produce a formulation suitable to include bearing measurements. Here we focus on verifying performance improvements w.r.t. the original formulation when bearing data is included.

The results show that the algorithms compete favorably with the benchmark algorithms. Despite that, issues regarding their computational demands and communications overhead must be dealt with, and are discussed in the thesis. We also discuss possible future directions of research to improve upon the work presented herein.

***Index Terms*—Network localization, distributed optimization, range-based localization, sensor fusion, ML estimation, bearing localization**

I. Introduction

A Localization Algorithm is a set of operations to perform upon available data in a network of vehicles in order to estimate their positions at a given instant. Most localization algorithms occur in two phases [1], in the **Measurement phase** information pertinent to the algorithm is collected, be it ranges, bearings, or connectivity data, then. In the **Localization phase** previously collected data is integrated via a series of operations, and the location estimates produced.

In general, in a network, there exists two types of points, nodes, whose location is unknown, and anchors, whose location is known.

Many classifications with which to differentiate Localization Algorithms exist in literature [2], [3], [4], we can distinguish them by their type of measurement, if they are range-based or range-free; by their method, be it optimization, or Kalman-like; by the network dynamics, or what assumptions are made for the motion of the nodes therein; and by whether they are single source, where a node's location is determined via its measurements with anchors only, or collaborative, where the estimate is obtained with the additional inter-node measurements.

A myriad of approaches to tackle the localization problem have been developed by the scientific community. Many of them are based on Kalman Filter, or its derivates, some examples include the single source scheme developed in [5] collaborative scheme in [6]. Optimization approaches include SDP (Semidefinite Programming) [7] and EDM (Euclidean Distance Matrix) completions [8]. Centralized approaches such as SDP and MDS (Multidimensional Scalling) [8] have been extensively worked by the scientific community, although the scalability of these methods is generally poor.

Distributed approaches have been explored due to their better properties in dealing with larger networks, in [9] the authors develop a gradient descent and a Gauss-Newton approach to tackle the problem, and show that the distributed solutions converge, under some assumptions, to their centralized equivalents. In [10], the problem is relaxed via convex underestimator terms instead of original non convex quadratic terms. In [11], the authors solve the exact optimization problem by way of a Minimization-Majorization scheme, with guarantees of convergence.

In our case, we develop a range-based, distributed, optimization-based, collaborative scheme, which integrates relevant information regarding node dynamics to yield position estimates, our approach mirrors closely the one taken in [11].

II. Framework

A. Network Notation

A network of nodes is represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, wherein $\mathcal{V}$ is the set of unknown node positions: $\{1, 2, \ldots, n\}$; $\mathcal{E}$ is the set of edges which link pairs of nodes (i.e., they can communicate with one another, and their relative range measurement is known to both) in the graph, an edge $i \sim j$ exists if node i has a link with node j. The set of nodes neighboring node i is denoted by $\mathcal{N}_i$. The set of points with known position, or anchors, is denoted by $\mathcal{A} : \{1, 2, \ldots, m\}$, each node i has an associated anchor set $\mathcal{A}_i$ which is a subset

of the original set $\mathcal{A}$, an edge between a node i and an anchor k is denoted as $i \sim k$.

A range measurement is denoted by d_{ij} between nodes i and j, or r_{ik}, between node i and anchor k.

In a similar vein, when dealing with bearing measurements, we assume a new graph, $\mathcal{G}_a = (\mathcal{V}, \mathcal{E}_a)$, where $\mathcal{V}$ again represents the set of nodes in the network, and $\mathcal{E}_a$, represent the set of edges between two nodes wherein a bearing measurement exists. Similarly to the range measurements case, an edge $s \sim u$ exists if node s has a bearing link with node u. The set of nodes neighboring node i in this case is denoted by $\mathcal{N}_i^a$. The set of anchors neighboring node i, which a bearing measurement is available, is a subset of $\mathcal{A}$, and denoted by $\mathcal{A}_i^a$.

The dimension of the space is denoted by d ($d = 2$ or 3). Furthermore, for the majority of this document, we assume that all the measurements relative to a given time-step are gathered simultaneously. Although simultaneous communications within all the edges of the network is unreasonable in underwater scenarios, we aim to develop an algorithm suitable for vehicle formations where it is reasonable to assume that the measurements between the elements of the network vary slowly over time and the effect of time skews between range measurements is thus moderate.

B. Measurement Noise Models

1) Ranges: Most measurements are contaminated by measurement errors. Throughout this work, it is assumed that range measurements are corrupted by an error term which is modeled as Gaussian Random Variable (GRV), for instance, the distance between two nodes i, j

$$d_{ij} = ||x_i - x_j|| + n_{ij}, \tag{1}$$

where $n_{ij} \sim \mathcal{N}(0, \sigma_r^2)$.

2) Bearings: In terms of the bearing measurement, we assume a von Mises-Fischer distribution [12], which can be looked at as the extension of the normal distribution in the circumference (2D) or the sphere (3D). In the case of a measurement between two nodes, we have the following pdf

$$p(\mu_{ij}|x_i, x_j) = C_p(k) e^{k \mu_{ij}^T \frac{\Delta x_{ij}}{|\Delta x|}}, \tag{2}$$

where μ_{ij} is the bearing measurement, coded as the noisy measurement of the unit vector of the direction $\Delta x_{ij} = x_i - x_j$. k is the concentration parameter, and $C_p(k)$, a normalization constant. In the 2D case, denoting the bearing angle between the two nodes as θ_{ij}, we can write the bearing measurement as $\mu_{ij} = \begin{bmatrix} \cos\theta_{ij} & \sin\theta_{ij} \end{bmatrix}^T$

Assuming furthermore that the angle θ is the true value corrupted by a GRV noise term, $\tilde{\theta}_{ij} \sim \mathcal{N}(0, \sigma_\theta^2)$.

The relation between k and and σ_θ^2, can be extracted via appropriate sampling [13], yielding $k = \frac{c}{\sigma_\theta^2}$, with c being a constant ($c \approx 90$).

C. Kinematics

The position of a node i and anchor k at a given point in time t is denoted by $x_i(t)$ and $a_k(t)$, respectively. It is assumed that velocity measurements, $v_i'(t)$, for every node, at any given time, is known. Ignoring the existence of current velocity, for simplicity, from a given initial position $p_i = x(t)$, a position $x_i(\tau)$ further on time can be written as follows

$$x_i(\tau) = p_i + \sum_{\zeta=t}^{\tau} v_i'(\zeta)\Delta T = p_i + v_i(\tau), \tag{3}$$

where $v_i(\tau) = \sum_{\zeta=t}^{\tau} v_i'(\zeta)\Delta T$.

III. Algorithm Derivation

A. DIeSEL

The principle followed is to solve the localization problem for a horizon spanning T sets of range measurements. Under the assumption of zero-mean, i.i.d., additive Gaussian range measurement noise terms, we reach via ML estimation to the following cost function, which we aim to minimize, where x is the stack of positions of the nodes within the time window T:

$$f(x(t)) = \sum_{\tau=t-T}^{t} \left(\sum_{i \sim j} \frac{1}{2} (||x_i(\tau) - x_j(\tau)|| - d_{ij}(\tau))^2 + \sum_i \sum_{k \in \mathcal{A}_i} \frac{1}{2} (||x_i(\tau) - a_k(\tau)|| - r_{ik}(\tau))^2 \right). \tag{4}$$

1) Reformulation: We adopt the approach in [11] and substitute the range terms by vectorial variables, whose Euclidean norm is constrained to be equal to the measurement, stacking the corresponding node-node y and anchor-node y ranges terms throughout the time window gives rise to two constrained sets: $\mathcal{Y} = \{y_{ij}(\tau) : ||y_{ij}(\tau)|| = d_{ij}(\tau),\ \forall\tau\ \forall i \sim j\}$, and $\mathcal{W} = \{w_{ik}(\tau) : ||w_{ik}(\tau)|| = r_{ik}(\tau),\ \forall\tau\ \forall i\ \forall k \in \mathcal{A}_i\}$. We define the design variable as $z = (x, y, w)$, constrained to the set $\mathcal{Z} = \{z = (x, y, w) :\ y \in \mathcal{Y},\ w \in \mathcal{W}\}$, for simplicity.

Exploring the kinematics we notice that

$$x_i(\tau) - x_j(\tau) = p_i - p_j + \Delta v_{ij}(\tau), \tag{5}$$

with $\Delta v_{ij} = v_i(\tau) - v_j(\tau)$, and

$$x_i(\tau) - a_k(\tau) = p_i - a_k(\tau) + v_i(\tau). \tag{6}$$

Stacking the $\Delta v_{ij}(\tau)$ and $-a_k(\tau) + v_i(\tau)$ appropriately on stacks Δv and α, respectively, and using the initial positions p stack instead of x, setting $z = (p, y, w)$, we reach the following matrix form of the cost function:

$$f(z) = \frac{1}{2} \left|\left| \begin{bmatrix} DA & -I & 0 \end{bmatrix} \begin{bmatrix} p \\ y \\ w \end{bmatrix} + \Delta v \right|\right|^2 + \frac{1}{2} \left|\left| \begin{bmatrix} E & 0 & -I \end{bmatrix} \begin{bmatrix} p \\ y \\ w \end{bmatrix} + \alpha \right|\right|^2, \tag{7}$$

where A is the Kronecker product between the arc-node incidence matrix associated with graph $\mathcal{G}$, and the identity matrix (with dimension equal to the space dimension), D is a tall concatenation of identity matrices, and E is a matrix of ones and zeros, which selects the appropriate p element to match the α element.

Expanding the norms, and discarding the constant terms, we reformulate finally our minimization problem to the following:

$$\begin{aligned} \underset{z}{\text{minimize}} \quad & f(z)_{=} z^T M z - b^T z \\ \text{subject to} \quad & z \in \mathcal{Z}, \end{aligned} \tag{8}$$

with

$$M = \begin{bmatrix} A^T D^T D A + E^T E & -A^T D^T & -E^T \\ -DA & I & 0 \\ -E & 0 & I \end{bmatrix}, \tag{9}$$

and

$$b = \begin{bmatrix} E^T \\ 0 \\ -I \end{bmatrix} \alpha - \begin{bmatrix} A^T D^T \\ -I \\ 0 \end{bmatrix} \Delta v. \tag{10}$$

2) Optimization: Although problem (8) is an optimization problem constrained to a nonconvex set, its cost function is quadratic. Therefore it admits a Lipschitz continuous gradient, the following result holds around a given point $z(\kappa)$

$$f(z) \leq f(z(\kappa)) + \nabla f(z(\kappa))^T (z - z(\kappa)) + \frac{L}{2} ||z - z(\kappa)||^2, \tag{11}$$

where L is the Lipschitz constant, it was determined to be $L = T[2\delta_{a_{max}} + |\mathcal{A}|_{max}]$, where $\delta_{a_{max}}$ is the maximum connectivity degree of the network $\mathcal{G}$, and $|\mathcal{A}|_{max}$ is the highest number of neighboring anchors for any given node in the network. The expression (11) majorizes the cost function, the strategy then followed to tackle the optimization problem is that of Majorization-Minimization (MM), as detailed in [14]: at each iteration, the cost function is majorized, then, this majorized function is minimized. The MM strategy employed ensures that the value of the objective function is non-increasing along the iterations. The result of minimization at each iteration, taking into account the result in expression (11), is an iteration of the projected gradient method on a non convex set, and it is guaranteed to converge to a stationary point [15]

$$z(\kappa + 1) = P_{\mathcal{Z}} \left(z(\kappa) - \frac{1}{L} \nabla f(z(\kappa)) \right), \tag{12}$$

where $P_{\mathcal{Z}}(z)$ denotes the projection of z onto the set $\mathcal{Z}$.

Decoupling the variables, we reach the distributed formulation of DIeSEL, shown in algorithm 1.

Algorithm 1 DIeSEL algorithm

Input: L // Lipschitz constant;
$\beta_i = \frac{L - T(\delta_i + |\mathcal{A}_i|)}{L}$;
$\{d_{ij}(\tau) : \forall i \sim j \; ; \forall \tau \in]t - T, T]\}$;
$\{r_{ik}(\tau) : \forall i \in \mathcal{V} \forall k \in \mathcal{A}_i \; ; \forall \tau \in]t - T, T]\}$;
$z = (p, y, w)$; // initialization
Output: $\hat{z} = (p, y, w)$; // estimate
while some stopping criterion is not met, each node i **do**
  broadcast $p_i(\kappa)$ to all neighbors
  $p_i(\kappa + 1) \leftarrow \beta_i p_i(\kappa) + \sum_{j \in \mathcal{N}_i} \sum_{\tau = t - T_0}^{t} (p_j(\kappa) + C_{i \sim j, i} y_{ij}(\tau, \kappa)) + \sum_{j \in \mathcal{A}_i} \sum_{\tau = t - T_0}^{t} (a_k(\tau) + w_{ij}(\tau, \kappa) + v_i(\tau))$
  for all $i \sim j$ **do**
    for $\tau = [t - T_0, t]$ **do**
      $y_{ij}(\tau, \kappa) \leftarrow P_{\mathcal{Y}_{ij}(\tau)} \left(\frac{L-1}{L} y_{ij}(\tau, \kappa) \right.$
      $\hookrightarrow \left. + \frac{1}{L} C_{i \sim j, i} (p_i(\kappa) - p_j(\kappa) - \Delta v_{ij}(\tau)) \right)$
    end for
  end for
  for all i **do**
    for all $k \in \mathcal{A}_i$ **do**
      for $\tau = [t - T_0, t]$ **do**
        $w_{ik}(\tau, \kappa) \leftarrow P_{\mathcal{W}_{ik}(\tau)} \left(\frac{L-1}{L} w_{ik}(\tau, \kappa) \right.$
        $\hookrightarrow \left. + \frac{1}{L} (p_i(\kappa) - a_k(\tau) - v_i(\tau)) \right)$
      end for
    end for
  end for
  $\kappa \leftarrow \kappa + 1$
end while

B. DIeSEL-B

Assuming moreover that all bearings are i.i.d. von Mises-Fisher random variables, including them in the ML formulation alongside the ranges yields the following cost function to be minimized

$$\tilde{g}(x) = f(x) - c \frac{\sigma_r^2}{\sigma_\theta^2} \tilde{h}(x), \tag{13}$$

where f is the original function obtained via ML (4), $\tilde{h}$ is defined as

$$\tilde{h}(x) = \sum_{\tau = t - T}^{T} \left(\sum_{s \sim u} \mu_{su}^T(\tau) \frac{x_s(\tau) - x_u(\tau)}{||x_s(\tau) - x_u(\tau)||} + \sum_i \sum_{k \in \mathcal{A}_i^a} \mu_{ik}^T(\tau) \frac{x_i(\tau) - a_k \tau}{||x_i(\tau) - a_k(\tau)||} \right). \tag{14}$$

1) Relaxation: The terms of the function $\tilde{h}$ are nonlinear due to the appearance of the norm function in the denominators. With the assumption that under low range (and bearing) measurement noise levels, the quadratic terms corresponding to ranges of the original formulation tend to small values in the optimum of the minimization problem, and the edge variables fit reasonably on the lines that connect nodes to nodes, for y

terms, or nodes to anchors, for w terms. We can approximate the $x_s(\tau) - x_u(\tau)$ and $x_i(\tau) - a_k(\tau)$ terms by, respectively

$$\begin{cases} x_s(\tau) - x_u(\tau) \approx y_{su}(\tau), \\ x_i(\tau) - a_k(\tau) \approx w_{ik}(\tau). \end{cases} \tag{15}$$

We notice that the relaxation is linear w.r.t. to y and w terms. The final minimization problem is trivial to reach:

$$\begin{aligned} \underset{z}{\text{minimize}} \quad & g(z) = z^T M z - b_+^T z \\ \text{subject to} \quad & z \in \mathcal{Z}, \end{aligned} \tag{16}$$

where M and b were carried from the DIeSEL formulation, b_+ is defined as:

$$b_+ = b + \psi b_a, \tag{17}$$

where $-b_a = \begin{bmatrix} 0 & \Phi_n & \Phi_a \end{bmatrix}^T$ stacks the measured bearings for the y and w stack elements appropriately, and $\psi = -c\frac{\sigma_r^2}{\sigma_\theta^2}$.

Under the same procedure used in DIeSEL, we can easily reach the distributed formulation for DIeSEL-B. Relevantly, since the M matrix remains constant, the Lipschitz constant L result is carried from DIeSEL. DIeSEL-B's algorithm is almost identical to DIeSEL's, with two alterations:

1) It takes the bearing measurement inside the time window;
2) The terms $+\frac{\psi}{L}\frac{\mu_{ij}}{d_{ij}}$ and $+\frac{\psi}{L}\frac{\mu_{ik}}{d_{ik}}$ are added to lines 12, and 18, respectively, provided the respective measurement exists, inside the projection functions.

C. Stopping Criterion

In terms of the stopping criterion, two modalities exist:

1) Max Iterations: The algorithm stops after a preset number of iterations;
2) Minimum Improvement: The cost function improves by less than a preset value.

We opt for the first option. Since we are in the context of a distributed algorithm, the number of iterations necessary to solve the problem is discussed in the results, in Section IV-B5.

D. Initialization of the algorithms

Since DIeSEL and DIeSEL-B solve a nonconvex optimization problem, they require a proper initialization in order to avoid local minima issues. We adopt the following strategy: we initialize the first estimates with low initialization noise σ_{init} and run the first T instances of the algorithm with this initial estimate, augmenting the time window until it reaches the desired T. After reaching T, we use the past locations estimated by the algorithms to initialize the design variables, Figure 1 exemplifies this scheme.

IV. Simulation Results

A. Setup

The software used for the folllowing simulations was 64-bit version MATLAB (R2012a). All the simulations were run on servers provided by ISR-IST, using 24-core processors with 64 GB of RAM memory. We use the centralized version of the algorithm for these simulations.

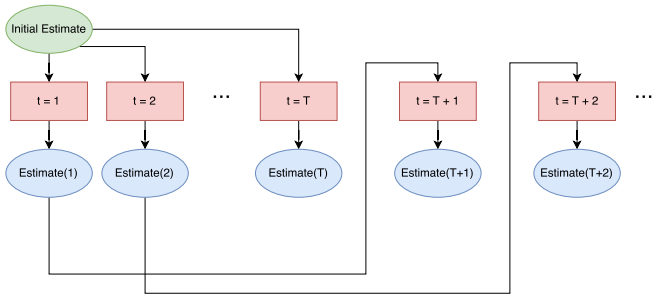

Fig. 1. Initialization scheme for the algorithms. The initial estimate, in green, is used to initialize the algorithm, in red, for the first T instances. Afterwards, the estimates produced by the algorithm, in blue, are used sequentially for future initializations of the algorithms.

1) Trajectories: We use the lap trajectory (the network trajectory is represented in Figure 2) throughout this report, it has a blend of linear and curved tracks, and it is representative in terms of results of other types of generic trajectories, such as lawnmower and helix curves.

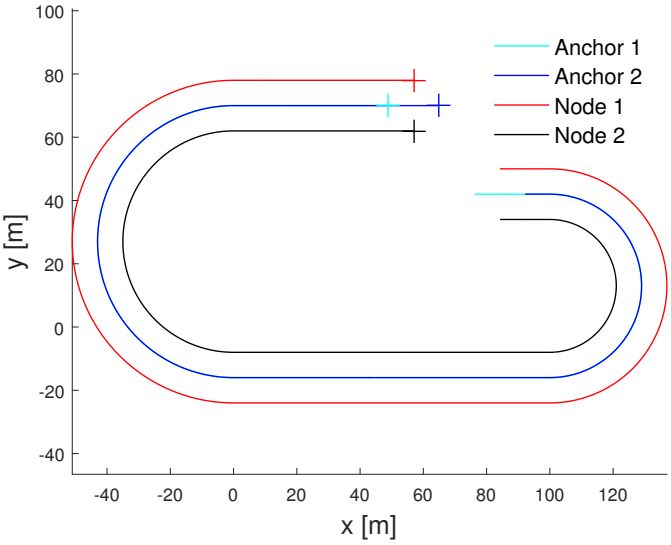

Fig. 2. Lap trajectory for the network formed by 2 nodes and 2 anchors. The nominal speed of the network is 1m/s, the speed on the curved tracks is altered depending to the curvature of the track in order to maintain formation. Network is fully connected w.r.t. to range measurements.

2) Performance Parameters: Two main parameters with which to assess the performance of the algorithms developed and the benchmarks are used:

1) Mean Error per Run: it is the mean of all estimation errors, per point, occurring during a run: $E = \frac{1}{N}\frac{1}{T_t}\sum_{\tau=1}^{T_t}\sum_{i=1}^{N}||x_i(\tau) - \hat{x}_i(\tau)||$
2) The mean of the former parameter along the various Monte Carlo runs is termed Mean Error: $M_E = \frac{1}{MC}\sum_{k=1}^{MC} E(k)$
3) Mean Error per Trajectory Point: it is the mean of all estimation errors, per point, per instant, occurring during the various Monte Carlo runs: $e(\tau) = \frac{1}{MC}\frac{1}{N}\sum_{k=1}^{MC}\sum_{i=1}^{N}||x_i(\tau) - \hat{x}_i(\tau, k)||$

Where T_t is the size, in terms of time-steps, of the trajectory, in our case, $T_t = 400$.

B. DIeSEL

1) Comparison with benchmarks: Three benchmark algorithms were used to assess the performance of the algorithms developed, Static Localization (SL, retrieved from [11]), and Extended Kalman Filter, EKF, and Monte Carlo Localization, MCL, based of the particle filter. The latter two algorithms were adapted from [16]. We present only the results for a single trajectory type, presented at Figure 2. We use the centralized version of DIeSEL (and DIeSEL-B) for all the simulation runs.

We present the plots for the mean error per trajectory point value e, and the CDFs for the mean error per run values e. The results are obtained from 100 Monte Carlo runs. We use $T = 8$, $\sigma_r = 0.5\text{m}$. We furthermore add a multiplicative gaussian noise term for the velocity measurements: $v_{\text{measured}} = (1 + \tilde{v})v_{\text{true}}$, with $\tilde{v} \sim \mathcal{N}(0, \sigma_v^2)$, and set $\sigma_v = 0.01$. We also add an error term on the initialization of the nodes $x_{\text{exact}}(0) - \hat{x}(0) \sim \mathcal{N}(0, \sigma_{\text{init}}^2 I)$, and set $\sigma_{init} = 2$. These last two parameters remain constant throughout the simulations.

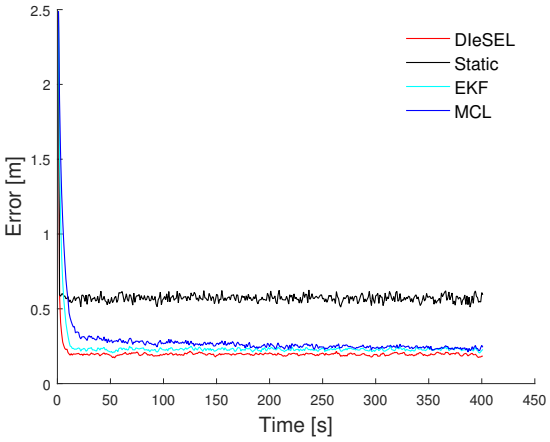

Fig. 3. DIeSEL mean error per trajectory point e for the laps trajectory. Average of 100 Monte Carlo runs. SL has the worst behavior of the comparison algorithms and converges the fastest, DIeSEL, has the best performance, and the second fastest convergence rate. EKF and MCL track closely behind DIeSEL, with larger transients periods, MCL being the slowest to converge.

We verify that under the conditions simulated, DIeSEL outperforms the other algorithms in terms of steady-state mean error values in all trajectories. The worst performing algorithm in this set of simulations is the SL algorithm, this algorithm is essentially DIeSEL constrained to a time window of $T = 1$. The dispersion of the mean error per run values is similar across the algorithms, as visible in the CDF curves, except for MCL. We conjecture that, due to the stochastic nature of the algorithm, since we initialize the particles around the initial guess, and not centered on the true position, the speed of convergence of the runs in MCL is rather variable.

Both the EKF and MCL algorithms have similar performances compared with DIeSEL, the successful employment

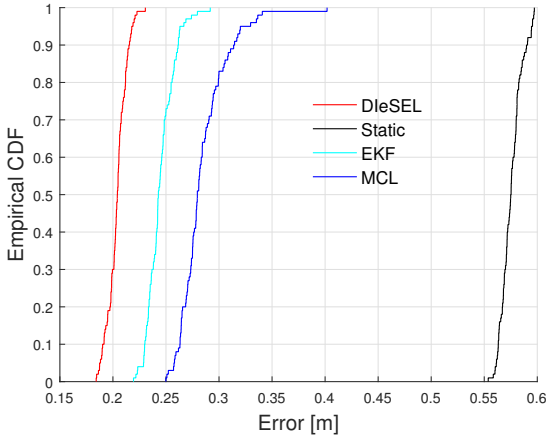

Fig. 4. DIeSEL CDF curves of mean errors per run (E) of the 100 trials performed for the laps trajectory. DIeSEL, EKF and SL have similar dispersion patterns. MCL results are the most dispersed.

of these algorithms hinges on prior tuning of their parameters, which can be time-consuming, especially for EKF. Furthermore, stability in EKF is not guaranteed, and the stochastic nature of MCL does not offer convergence guarantees for all its particles.

For a general idea of the dispersion of the estimates of DIeSEL, we show in Figure 5 a sample estimated lap trajectory, zoomed on a particular representative regions. We plot the respective SL estimated trajectory alongside DIeSEL to serve as a benchmark.

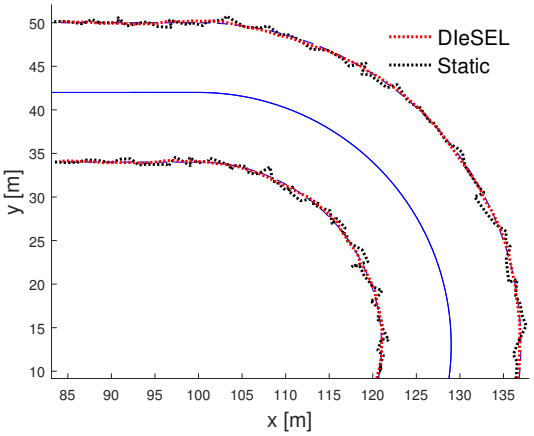

Fig. 5. Estimated node positions in the lap trajectory, observe that the DIeSEL trajectory is noticeable smoother than the SL counterpart, which suffers from the additional drawback of registering inversions of movement.

2) Sensitivity to σ_r and T: We wish now to study the influence of the values of the range measurement noise standard deviation σ_r and the time window T on the performance of DIeSEL. For that effect, we run simulations varying one of the variables at a time. The reference values for the two sets of simulations are $\sigma_r = 0.5\text{m}$ and $T = 5$ or $T = 8$. We vary σ_r from 0.5m to 3m (approximately 5 to 30% of the minimum range measured in the network). We run simulations for the following set of T: 1, 2, 3, 5, 10 and 20. For each set of

simulations, we run 100 Monte Carlo trials. We plot in Figures 6 and 7 the mean error M_E values for the set of simulations on σ_r and T, respectively.

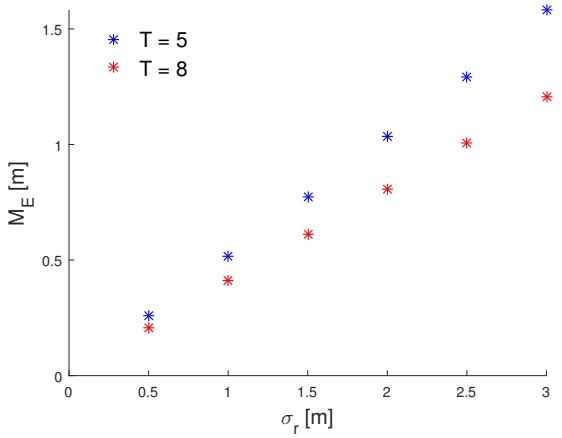

Fig. 6. DIeSEL mean errors M_E for the lap trajectory, blue for T = 5, red for T = 8, varying range measurement standard deviation σ_r . Average of 100 Monte Carlo runs. M E increases linearly with σ_r. The slope is milder at the larger time window.

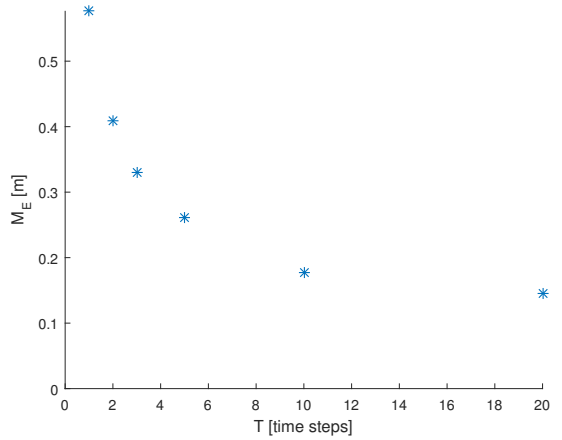

Fig. 7. DIeSEL mean error M_E values for lap trajectory, with different time window sizes. Average of 100 Monte Carlo Trials. The M_E values have a inverse proportional relation with the time window sizes.

As observed in Figure 6, the performance of DIeSEL, as for any range-based localization algorithm, worsens with the increase of the range measurement noise levels. Furthermore, the relation seems linear, but we do not expect for the slope to be constant w.r.t. the time window T. We expect the slope to be milder with larger time windows, since it integrates a larger set of measurement data, as shown in the smaller slope of the red data points ($T = 8$, $\frac{M_E}{\sigma_r} \approx 0.4$) compared to the blue ones ($T = 5$, $\frac{M_E}{\sigma_r} \approx 0.5$).

As we can infer from Figure 7, the performance of the algorithm betters indeed with the increase in the time window T, this is due to the fact that more information is available in larger windows, thus, the precision of the estimates augments with T. The behavior is nevertheless not linear, as more frames of information (data from a single sampling period) are included, the additional performance return is ever more slight, thus an inverse proportional relation is observed between mean error and T.

3) Effect of asynchronous or missing measurements: We made the assumption during DIeSEL's formulation that all the measurements would arrive simultaneously, this is not a correct assumption, especially in the context of underwater communications, in these environments, solutions like TDMA [17] (Time Division Medium Access) are used. For this effect we test the lap trajectory, with the reference parameters of the previous section ($T = 5$, and $\sigma_r = 0.5\text{m}$), using a Sample and Hold (S&H) scheme in which the information regarding each node is updated individually, in alternating time-steps. We plot the mean error per trajectory point e for this set of experiments in Figure 8, over 100 Monte Carlo trials. Another assumption made was that all the measurements would be available at every time-steps, in real life, this cannot be fully guaranteed, packets of data may arrive with errors, or not arrive at all [18]. For that effect, we introduce a new parameter p_L, probability of failure in transmission, and vary it from 0 to 50%, in increments of 10%, we hold a measurement if its updated value is lost. We perform 100 Monte Carlo trials per p_L value, and plot the mean error values for each p_L in Figure 9.

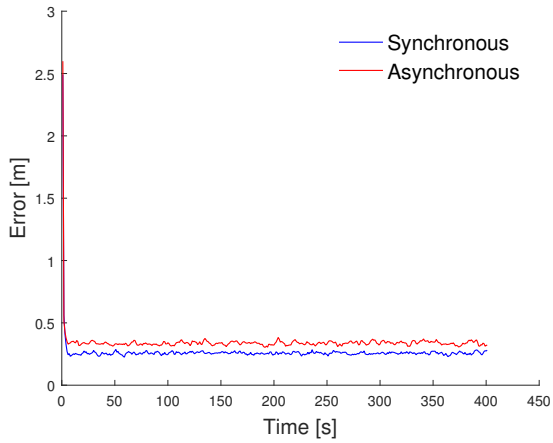

Fig. 8. DIeSEL mean error per trajectory e values plots for the reference synchronous (blue) and asynchronous (red) schemes. Average of 100 Monte Carlo trials. The errors are lower in the synchronous case.

Firstly, we observe an expected result, the asynchronous (A) case does indeed perform worse than the synchronous (S) one. (A) has effectively 17 (5 times the node-node measurements, 3 times the anchor-node measurements) range measurements inside the time window, whereas (S) has access to almost the double, at 30 range measurements. The betterment in terms of performance from (A) to (S) is then akin to the behavior observed in the time window influence in DIeSEL analysis in Section IV-B2: the more information the algorithm possesses, the higher the accuracies it achieves.

We can observe a similar dynamic on the results in Figure 9. The performance worsens with increasing probability of failure of transmission p_L, this is to be expected, The accuracy of the measurements is linked to the effective amount of

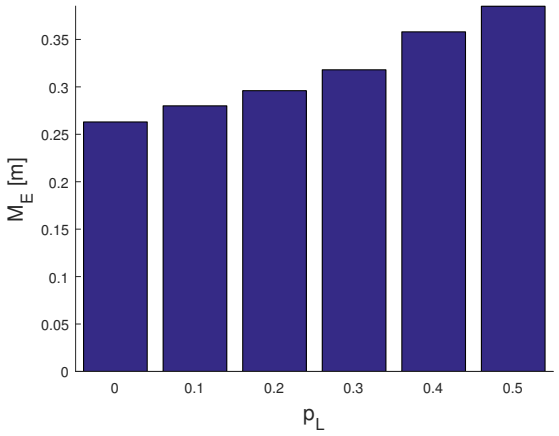

Fig. 9. DIeSEL mean error M_E values for each value of probability of failure of transmission p_L. Average of 100 Monte Carlo trials. The errors grow as the rate of failure increases.

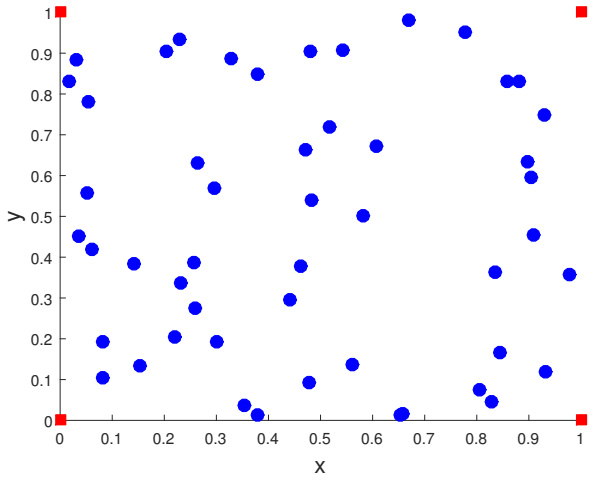

Fig. 10. Large network configuration. Anchors (4), at the corners of the unit square, at red, nodes (50) in blue.

measurements, and not entirely on the its quantity *per se*. As the effective amount of measurements decreases with p_L, the mean error values M_E augment.

Nevertheless, we can conclude that the existence of asynchronous or missing measurements during the operation of DIeSEL does not hinder drastically its performance, given the assumption that the ranges and velocities evolve slowly over time and using the aforementioned S&H scheme as a mechanism of tackling the asynchronous missing transmission issues. We would expect higher error if the network shape were to change rapidly in a maneuver, or if a higher diversity were to be observed in terms of velocities of the vehicles in the network.

4) Scalability: In this section we aim to study the effect of having a larger amount of points in the network. We perform simulations with a larger network, in order to determine if the same order of precision is obtained with larger scale networks. The sample network is plotted in Figure 10, the units are adimensional, we hence normalize the results with the range measurement noise standard deviation σ_r in order to assess the performance of DIeSEL in this set of simulations.

For the simulations, we set speed at zero, we are essentially using snapshots of the network in order to determine its node location estimates. We vary the node connectivity radius c_r, or the maximum distance from which two points are considered neighbors, c_r take values of 0.2, 0.25 and 0.3. The average neighbor count, AEC, per nodes in the network in each scenario is show in Table I.

TABLE I
AVERAGE NEIGHBOR COUNT, AEC, FOR THE NODES OF THE NETWORK WITH EACH CONFIGURATION OF CONNECTIVITY RADIUS c_r.

c_r	0.2	0.25	0.3
AEC	4.2	6.6	9.1

We run 100 Monte Carlo trials for each connectivity radius, we set range measurement noise standard deviation $\sigma_r = 0.05$ (5% of the square side, or 30% of the maximum possible range). Time window size T, is set at 5, the duration of the static trajectory is 40 time-steps. The points are initialized with error, the initialization noise standard deviation is set at $\sigma_{\text{init}} = 0.02$ (or normalized $\frac{\sigma_{\text{init}}}{\sigma_r} = 0.4$). We plot the normalized mean error values, $\frac{M_E}{\sigma_r}$, for each scenario, in Figure 11.

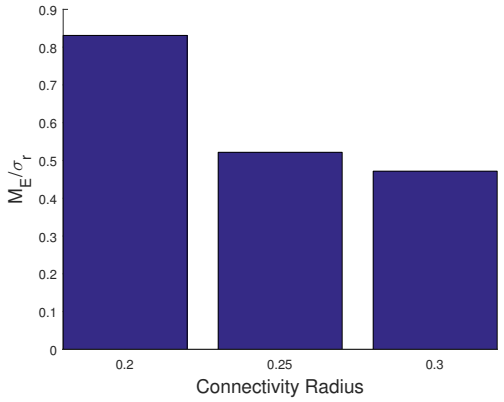

Fig. 11. DIeSEL normalized mean error values, $\frac{M_E}{\sigma_R}$ for each value of connectivity radius c_r, the tendency is for $\frac{M_E}{\sigma_r}$ to decrease with c_r. Recall that the reference value for the network in the lap trajectory for the same time window was 0.4. The large decrease from the first to the second scenario may be due to the lack of edges in some of the nodes in the former.

As we can observe, the tendency in the decrease of the normalized mean error $\frac{ME_E}{\sigma_r}$ w.r.t. the connectivity radius c_r is to be expected. As c_r increases, so does the average number of neighbors per node, as more edges are added to the estimation process, the estimates should become more precise.

Furthermore, the ratio $\frac{M_E}{\sigma_r}$ remains on the same order of magnitude in these set of simulations as those observed in the range measurement noise analysis σ_r performed in Section IV-B2. Recall, that for $T = 5$, the ratio $\frac{M_E}{\sigma_r}$ observed therein was around 0.5. The values obtained in this section remain similar, but it is necessary to point out that the average number of neighbors per node in the former simulations was

3, below the values obtained here. This dynamic is particularly surprising when we consider that each node the third scenario ($c_r = 0.3$) has on average three times the number of neighbors, but around the same $\frac{M_E}{\sigma_r}$ as the earlier case. A possible explanation is that due to the relatively shorter distances between some set of nodes, in conjunction with an improper initialization, and the high $\frac{\sigma_r}{c_r}$ ratio, leads frequently to the estimates getting 'stuck' on wrong local minima. The sharp decrease from the first to the second scenarios ($c_r = 0.2$ to $c_r = 0.25$) can probably be explained in a similar way. We can observe in Figure 12, which plots the measurement edges in the network with $c_r = 0.2$ that there exist several nodes in the network with two or less neighbors, the ambiguity towards their location due to this fact leads for the first scenario to be particularly vulnerable regarding both initialization, and range measurement noise.

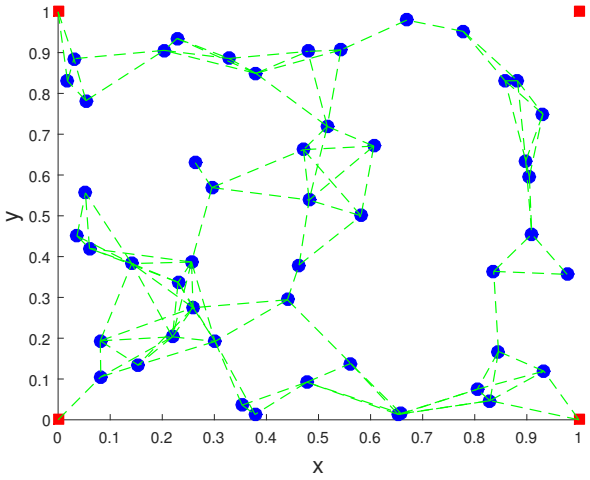

Fig. 12. Graphs representing connectivity status of the network with several connectivity radius $c_r = 0.2$. Anchors (4) in red, nodes (50) in blue, and range measurement edges in dashed green.

5) Time Performance: In terms of time expenditure of DIeSEL, it is dependent of the amount of iterations chosen, since it is unfeasible to opt for minimum improvement strategies, since, in a distributed sense, all a node has is access to its own information, and not the global objective function. For that effect, we simulate various MC runs (20), with the lap trajectory, we set minimum improvement as our stopping criterion to 10^{-5} (experiments showed that the performance results do not vary significantly with use of lower values), and record two quantities, the mean iterations per time step M_{it} (Figure 13), and the mean time expenditure per time step M_t (Figure 14), varying the time window, for DIeSEL. The M_t values for the benchmarks are present in table II.

The increase of iterations needed for each time window observed in Figure 13 is expected since the the step-size for the used projected gradient method is $\frac{1}{L}$, and L is proportional to the time window size. The quadratic behavior of the data points in Figure 14 is a confluence of the effect of increasing the iteration numbers, and the increase in the edge variables (y and w), with larger time windows.

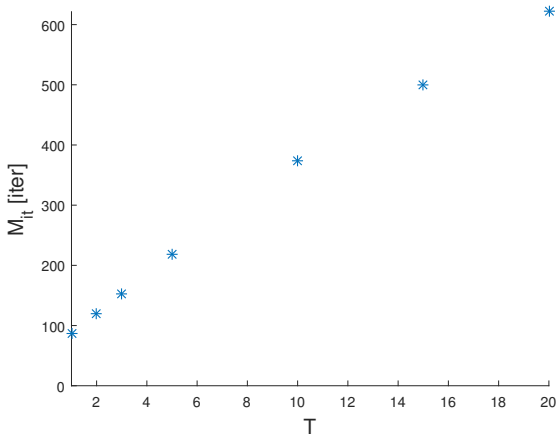

Fig. 13. DIeSEL mean iterations per time step M_{it} values, required at each time window configuration. Averaged over 20 Monte Carlo runs. The tendency of M_{it} appears to be roughly linear w.r.t. the time window T.

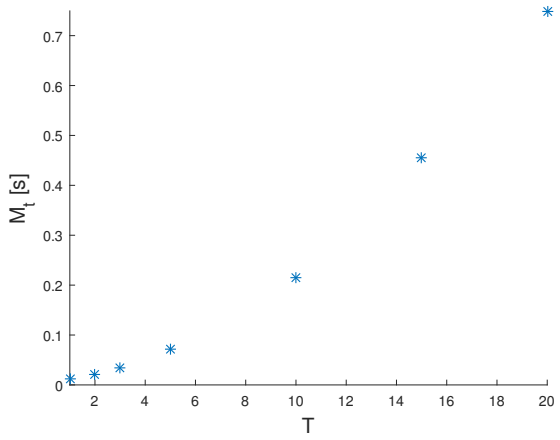

Fig. 14. DIeSEL mean time expenditure per time step M_{t} values, required at each time window configuration. Averaged over 20 Monte Carlo runs. The tendency, in this case, of M_{t} appears to roughly quadratic w.r.t. to the time window T.

TABLE II
MEAN TIME EXPENDITURE M_{t} VALUES FOR DIESEL'S BENCHMARKS. AVERAGE OF 20 MC. EKF IS THE FASTEST ALGORITHM, FOLLOWED BY MCL, LS TRAILS BEHIND IN LAST.

Algorithm	SL	EKF	MCL
M_t [s]	4.7×10^{-2}	4.7×10^{-4}	4.2×10^{-3}

Observing the disparity of mean time expenditures of EKF and MCL relatively to DIeSEL make us consider that these simpler implementations may be preferred in mission, DIeSEL can be used as a post-mission processing tool in order to yield more precise location estimates.

Recall also that we are using the centralized version of the algorithm in this set of simulations. From DIeSEL distributed algorithm 1, provided that the amount of average neighbors of a node in a network remains constant, we would also expect for the time expenditure per node, assuming each one has an independent processing unit, to grow quadratically with the

time window increase, since the both the amount of iterations and the edge variables augment in tandem with the time window T. We would expect for the time expenditure per node to remain constant, if the amount of elements in the network grows, while maintaining the same average number of neighbors per node, using the same time window T.

C. DIeSEL-B

We ran an initial set of simulations with DIeSEL-B, using the lap trajectory, $T = 5$ and $\sigma_r = 0.5\text{m}$, using the value of $\psi \approx 90\frac{\sigma_r^2}{\sigma_\theta^2}$ obtained via ML estimation. DIeSEL-B would perform better than DIeSEL with low angle noise levels $\sigma_\theta \leq 5^\text{o}$, after this value, it would perform worse. Further investigation lead to the conclusion that the assumptions made in (15) would be poorer the higher the angle noise standard deviation. As a temporary workaround, we use a low angle standard deviation $\sigma_\theta = 1^\text{o}$, using the original ψ.

We consider the original network in the lap trajectory, we progressively add bearing measurement edges to its network:

(I) No bearing edge added;
(II) Bearing edge between the two nodes added;
(III) The previous scenario plus each node creates an edge with one of the anchors;
(IV) Network fully connected w.r.t. to bearings.

We present the mean errors M_E for each scenario on Figure 15. As we can see the addition of bearing measurement edges in DIeSEL-B lowers the estimation errors, as it is to be expected when adding further information beyond the range measurements required by DIeSEL

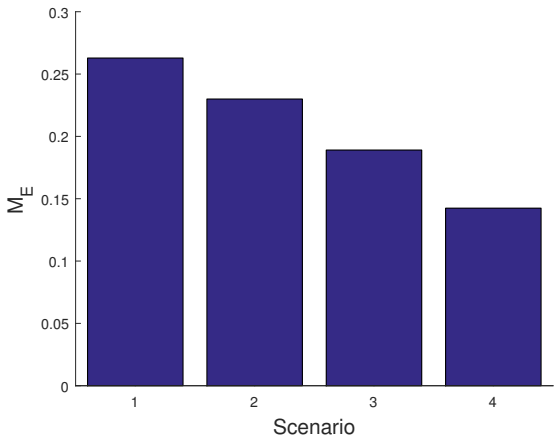

Fig. 15. DIeSEL-B mean error M_E values for the lap trajectory, for the various scenarios. Average over 100 Monte Carlo runs. The addition of bearing edges on scenarios II-IV drives down the mean error values from I.

V. Discussion

In this work, the wireless sensor network localization problem was addressed via an optimization approach. The result was a range-based, distributed, provably convergent, and parameter-free algorithm, which solves an exact optimization problem, with competitive performance with other benchmarked strategies. Furthermore, an adaptation of DIeSEL, via a relaxation, produced DIeSEL-B, a scheme suitable to include bearing information as a complement with ranges, with improvements in performance compared with the former, under certain conditions.

A major advantage of DIeSEL w.r.t. classical approaches such as EKF or MCL, besides its convergence guarantees, is its lack of design parameters, its only parameter T, is not critical and can be chosen according to the situation. It would be then tempting to consistently use high T values. Several drawbacks are apparent, first, the use of such large time windows increases the amount of communications necessary amongst the nodes to properly solve the optimization problem between themselves, which could be unfeasible.

On the topic of necessary communications, given the amount of iterations needed to solve for each new set of data, we do not foresee DIeSEL being competitive with less expensive, in terms of communication cost, methods such as the EKF or MCL solutions with which it was benchmarked. It is not thus, in its distributed state, an viable for online location estimation in scenarios such as underwater environments. Although its centralized version could potentially be used online, provided that the time windows chosen are not very large, and the number of sensors not very numerous (around the number used in the simulations).

Nevertheless, DIeSEL may be used as a post-processing tool. This can be particularly useful in applications such as marine prospection, where precise location estimates are fundamental to integrate data from other sensors, but where the environment is also prohibitive in terms of communications, and the vehicles have to rationalize their energy expenditure: a less powerful, but more inexpensive solutions can be used to navigate whilst the network of vehicle is in mission; then, DIeSEL can be used afterwards to produce to estimate the past trajectories with the all the collected range and speed data.

Regarding velocities, the model used is rather simple, we did not address the low pass behavior of most velocity measuring devices. In our case, we justify the use of the present model due to the assumptions w.r.t. the behavior of the network of vehicles, namely, the speeds are assumed to be more or less constant, and the movement direction of the whole network changes slowly along time. Moreover, for simplicity, we assumed a stationary fluid scenario, therefore the velocity measurements w.r.t. to the fluid were equal to the inertial values, in real life scenarios, this proposition does not generally hold. In the course of this work, we developed some formulations of DIeSEL which took into account the existence of current values, one centralized, which assumed a global current velocity, and another distributed, wherein all the nodes have decoupled current values (which is an incorrect assumption). The performance of these formulations was not satisfactory and further work is necessary in this respect to make DIeSEL a truly viable alternative to existing solutions. The integration of a current value should be the main focus for future research.

Further improvements to DIeSEL could include the integration of other available information, for example, if the

relative disposition of the network is known beforehand this information can be used to provide proper initialization, in order to avoid situations in which the estimated positions become 'stuck' in local minima.

An advantage of DIeSEL-B is that the bearing information is integrated linearly, reusing range measurement edge variables, bypassing typical non-linearities associated with angle measurement and trigonometric functions in an EKF formulation, for example. A major drawback of this algorithm stems from the relaxation. The integration of a bearing measurement demands a prior existence of a range measurement along the same edge, making the existence of standalone bearing measurements unfit for the algorithm, although in practice range measurements are more pervasive than their bearing counterparts, this issue can be a target of future work.

The use of an additional set of bearing data proved indeed that DIeSEL results could be improved by DIeSEL-B, even the marginal addition of bearing measurements aided to decrease mean estimation error values. The use of the relaxation revealed problematic in terms of its assumptions, which were not appropriate, specially at high bearing measurement noise levels, which lead to the conclusion that ψ would need to be a design parameter. No exhaustive study or simulations were performed in order to map the characteristic ψ, this also can be an object of future study. Another possible avenue of study is the use of directional distributions with more degrees of freedom for the 3D case, such as the Kent distribution [19], to account for different angle measurement precisions in that scenario.

REFERENCES

[1] S. Wang and H. Hu, "Wireless sensor networks for underwater localization: A survey, ces-521," *University of Essex*, p. 224, 2012.

[2] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 22, pp. 54–69, July 2005.

[3] G. Han, J. Jiang, L. Shu, Y. Xu, and F. Wang, "Localization algorithms of underwater wireless sensor networks: A survey," *Sensors*, vol. 12, no. 2, pp. 2026–2061, 2012.

[4] J. Bachrach and C. Taylor, *Localization in Sensor Networks*, pp. 277–310. John Wiley & Sons, Inc., 2005.

[5] P. Batista, C. Silvestre, and P. Oliveira, "Ges source localization based on discrete-time position and single range measurements," in *21st Mediterranean Conference on Control and Automation*, pp. 1248–1253, June 2013.

[6] H. J. Rad, T. van Waterschoot, and G. Leus, "Cooperative localization using efficient kalman filtering for mobile wireless sensor networks," in *2011 19th European Signal Processing Conference*, pp. 1984–1988, Aug 2011.

[7] Y. Ding, N. Krislock, J. Qian, and H. Wolkowicz, "Sensor network localization, euclidean distance matrix completions, and graph realization," *Optimization and Engineering*, vol. 11, pp. 45–66, Feb 2010.

[8] I. Dokmanic, R. Parhizkar, J. Ranieri, and M. Vetterli, "Euclidean distance matrices: A short walk through theory, algorithms and applications," *CoRR*, vol. abs/1502.07541, 2015.

[9] G. C. Calafiore, L. Carlone, and M. Wei, "Distributed optimization techniques for range localization in networked systems," in *49th IEEE Conference on Decision and Control (CDC)*, pp. 2221–2226, Dec 2010.

[10] C. Soares, J. Xavier, and J. Gomes, "Simple and fast convex relaxation method for cooperative localization in sensor networks using range measurements,"

[11] C. Soares, J. Xavier, and J. Gomes, "Distributed, simple and stable network localization," in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 764–768, Dec 2014.

[12] R. Fisher, "Dispersion on a sphere," *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 217, no. 1130, pp. 295–305, 1953.

[13] I. S. Dhillon and S. Sra, "Modeling data using directional distributions," tech. rep., University of Texas, 2003.

[14] D. R. Hunter and K. Lange, "A tutorial on mm algorithms," *The American Statistician*, vol. 58, no. 1, pp. 30–37, 2004.

[15] A. Beck and Y. C. Eldar, "Sparsity constrained nonlinear optimization: Optimality conditions and algorithms," *CoRR*, vol. abs/1203.4580, 2012.

[16] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[17] S. Climent, A. Sanchez, J. V. Capella, N. Meratnia, and J. J. Serrano, "Underwater acoustic wireless sensor networks: Advances and future trends in physical, mac and routing layers," *Sensors*, vol. 14, no. 1, pp. 795–833, 2014.

[18] R. Petroccia, C. Petrioli, and J. Potter, "Performance evaluation of underwater medium access control protocols: At-sea experiments," *IEEE Journal of Oceanic Engineering*, vol. PP, no. 99, pp. 1–10, 2017.

[19] J. T. Kent, "The fisher-bingham distribution on the sphere," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 44, no. 1, pp. 71–80, 1982.