Polynomial Time Approximation Algorithms for Localization based on Unknown Signals

Johannes Wendeberg, Christian Schindelhauer

Department of Computer Science, University of Freiburg, Germany
{wendeber, schindel}@informatik.uni-freiburg.de

Abstract. We consider the problem of anchor-free self-calibration of receiver locations using only the reception time of signals produced at unknown locations and time points. In our settings the receivers are synchronized, so the time differences of arrival (TDOA) of the signals arriving at the receivers can be calculated. Given the set of distinguishable time points for all receivers the task is to determine the positions of the receivers as well as the signal sources.
We present the first polynomial time approximation algorithms for the minimum problem in the plane, in which the number of receivers is four, respectively the number of signals is three. For this, we first consider the problem that the time points of m signals are jittered by at most some $\epsilon > 0$. We provide an algorithm which tests whether n given receiver positions are feasible with respect to m unknown sender positions with a run-time of $\mathcal{O}(nm^2)$ and we provide an algorithm with run-time $\mathcal{O}(nm \log m)$ which tests the feasibility of m given sender positions for n unknown sender positions. Using these tests, we can compute all possible receiver and signal source positions in time $\mathcal{O}\left(\left(\sqrt{2}/\epsilon\right)^{2n-3} n^2 m\right)$, respectively $\mathcal{O}\left(\left(\sqrt{2}/\epsilon\right)^{2m-3} nm \log m\right)$.

1 Introduction

The problem of location awareness of computing devices plays an important role in many engineering fields. A popular technique for acquiring the positions of devices is hyperbolic multilateration, as for example used in the LORAN and DECCA systems, global navigation satellite systems (GNSS), or cellular phone tracking in GSM networks.

In hyperbolic localization the position of a signal origin is located by a set of synchronized receivers. The times of arrival or time differences of arrival (TDOA) of a signal determine a system of hyperbolic equations. The benefit of this TDOA multilateration is that no control over the signal source is needed. Arbitrary sources of sound can be overheard, or the signal of radio emitters. Dedicated signal sources, as for example a tracked moving ultrasound beacon, can be primitive and cheap. It needs only to emit ultrasound pulses – no control signal is required.

In an extension the positions of anchors are *not* given a priori and are obtained as a result of the calculations. We present an approach to solve the problem in an approximation scheme yielding the best explanation of receiver locations for given time differences of arrival, up to a threshold of ϵ, which is chosen on the basis of the input error.

1.1 Related Work

The problem of TDOA localization has been widely researched and is present in literature. In conventional hyperbolic multilateration with fixed receiver positions the problem of signal localization is solved in closed form [4], [15], or by iterative approaches, and position estimates can be refined using Kalman filtering [5].

TDOA times are calculated by discrete timestamping [1], [6] or by cross correlation of audio streams [21]. Sources of information may be ultrasound signals [22], often in combination with RF signals. These are solved as *Time of Arrival* approaches [13], [16].

The problem of self-localization of receivers using only TDOA data and no anchor points is seldom considered in literature. In [11] a combination with DOA ("direction of arrival") data is presented as an initialization aid. In some cases the signals are assumed to originate from a large distance, the "far-field case" [6], [7], [20]. For near-field signals in the unit disc an upper bound of the error is shown in [17]. In [14] an approach is presented under the assumption that speakers are close to the receivers. In some contributions iterative algorithms are used to compute a solution [1], [22]. However, they tend to get stuck in local minima of the error function.

If many receivers are available there exists a closed form solution. For at least eight receivers in the plane, respectively ten receivers in three-dimensional space, the problem can be solved using matrix factorization [12].

Our refinement approach is inspired by branch-and-bound algorithms [9] usually applied to discrete problems. An algorithm for integer programming was first presented in [3]. By extension with Linked Ordered Sets it can be applied to nonlinear integer problems [8]. We saw some contributions suggesting the application of branch-and-bound to nonlinear nonconvex functions for global optimization [10]. However, problem solvers using branch-and-bound algorithms for multi-dimensional continuous space are rare.

2 Problem Setting

Let M_i $(1 \leq i \leq n)$ be a set of receivers and S_j $(1 \leq j \leq m)$ a set of signal origins, all positioned in the unit square of the Euclidean plane $\mathbb{R}^2$. Both, the receiver and the signal positions are unknown. The signals are emitted at unknown times u_j. Each signal S_j is propagated to each synchronized receiver M_i in a direct line with constant signal speed c and is detected at times t_{ij}. These times are the sole information given in the system. We normalize $c = 1$ for the simplicity and so, the signals and receivers and the event times satisfy the nm signal propagation equations for all $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, m\}$

$$t_{ij} - u_j = \|M_i - S_j\| \tag{1}$$

where $\| \cdot \|$ denotes the Euclidean norm. From these equalities we cannot derive absolute coordinates. We remove transitional, rotational and mirroring symmetries by placing the first receiver M_1 at $(0, 0)$, placing the second receiver on the x-axis and place the third receiver on the half-plane with positive y-coordinates.

The degrees of freedom $\mathcal{G}_2$ have been considered in [19] and [22]. Each of the nm equations decrements the degree of freedom. For n receivers and m signals we have $\mathcal{G}_2(n, m) = 2n + 3m - nm - 3$ degrees of freedom in the plane. If $\mathcal{G}_2(n, m) > 0$ then either infinitely many solutions or no solutions exist. Only if $\mathcal{G}_2(n, m) \leq 0$ the solution space can be reduced to a single solution. However, also no solutions, a constant number of solutions or infinitely many solutions may exist in this case, depending on the input. We conjecture that for $\mathcal{G}_2(n, m) < 0$ no side solutions exist if the input is derived from signal sources and receivers in general position, i.e. this conjecture holds with probability 1 for random positions from the unit square.

Definition 1. ***Anchorless localization problem*** *(self-calibration based on TDOA): Given the exact time points t_{ij} compute the positions of senders and receivers such that Equation (1) is satisfied.*

The problem is only solvable, if $n = 4, m \geq 5$, or $n = 5, m \geq 4$, or $n \geq 6, m \geq 3$. If the system is only slightly over-constrained, there is some chance of ambiguity. For example, for $n = 4, m = 5$ the number of solutions is bounded by 344 [19]. For $n = 6, m = 3$ at most 150 solutions exist. We have found some inputs where at least two solutions exist for $n = 4$ and $m = 5$. However, their exact number and probability is not known at the moment and are part of future work.

If the system is highly over-constrained, i.e. $n \geq 8$, $m \geq 4$ yielding $\mathcal{G}_2(n, m) \leq -7$, then Pollefeys et al. [12] showed how this nonlinear equation system can be transformed into a linear equation system, which allows for a polynomial time solution with run-time $\mathcal{O}(m^2 n^2)$. For other values only heuristic algorithms are known, which do not generate solutions for all inputs, as in [1] and [22].

Clearly, there are precision limits for the inputs t_{ij}. So, we consider the relaxed inequalities which assumes an error bound $\epsilon > 0$ for all $i \in \{1, \dots, n\}$, $j \in \{1, \dots, m\}$.

$$t_{ij} - \epsilon \leq u_j + \|M_i - S_j\| \leq t_{ij} + \epsilon \tag{2}$$

If we can compute a solution which satisfies these inequalities we call this an ϵ-approximation. Note that we are aware that the positions of the receivers or signals might be further than ϵ from the real position. Furthermore, in an ambiguous problem it is possible that we approximate multiple solutions which are not even close to the original positions. Then, we cannot decide which one is the correct solution. However, it is the best one can achieve given the error-prone problem setting.

Definition 2. ***Approximation problem of anchorless localization****: Given the time points t_{ij} and u_j and an error margin ϵ compute a possible set of positions of senders and receivers such that the inequalities (2) are satisfied, if they exist.*

This problem is not addressed by Pollefeys' algorithm. Furthermore, for small numbers of receivers ($n < 8$) or small numbers of signals ($m = 3$) no solution is known. We have presented a solution for large numbers of senders and receivers randomly distributed in a unit disk [17]. For synchronized receivers we can compute in time $\mathcal{O}(nm)$ an approximation of the correct relative positions within an absolute error margin of $\mathcal{O}\left(\frac{\log^2 m}{m^2}\right)$ with probability $1 - m^{-c} - e^{-c'n}$.

For small n and m there is a naïve solution for finding receiver and signal positions. For this one can test all $(2^{-\frac{3}{2}}\epsilon)^{3-2n-2m}$ positions of senders and receivers in a grid of cell size $\frac{1}{2\sqrt{2}}\epsilon$ whether they satisfy the inequalities (2). From the positions the signal time can be easily computed and the inequalities can be checked in time $\mathcal{O}(nm)$. When this exhaustive search has tested receivers and signals within the distance $\frac{1}{2}\epsilon$ from the correct solution this implies an overall error of ϵ, if there exists a solution.

Theorem 1. *The naïve approximation algorithm solves the approximation problem in time* $\mathcal{O}((2^{-\frac{3}{2}}\epsilon)^{3-2n-2m}nm)$.

Proof. Follows by testing all positions of the $2n + 2m - 3$ dimensional grid. Note that the constant three follows since we set M_1 to the origin and M_2 to the x-axis to reduce symmetries. The production time u_j of the signal can be computed from the distances, therefore we need to search only space and not time.

3 Polynomial Time Approximation for Small Numbers of Receivers

This approximation algorithm consists of two components. The first component tests for a given set of receivers if there exists a set of signal sources satisfying the constraints. The second component is a recursive tree search to find the receiver positions by applying the first test with increasing precision.

3.1 A Test for the Feasibility of Receiver Positions

If the receiver positions are known the location of the signal sources can be computed very efficiently. This observation inspires the following test algorithm. The problem of given receiver positions is to find signal sources which satisfy the inequalities (2).

We combine two receiver times t_{kj} and $t_{\ell j}$ for one signal source j to the time difference of arrival $\Delta t_{k\ell j} = t_{kj} - t_{\ell j}$ and yield the hyperbolic equation

$$\Delta t_{k\ell j} = \|M_k - S_j\| - \|M_\ell - S_j\| \tag{3}$$

for the exact solution which describes a hyperbola, called $\mathcal{H}(k, \ell, j, \Delta t_{k\ell j})$ with M_k and M_ℓ as focal points and S_j residing on the curve. For the approximation problem we consider the inequality

$$|\Delta t_{k\ell j} - (\|M_k - S_j\| - \|M_\ell - S_j\|)| \leq \epsilon \tag{4}$$

where ϵ is an upper bound of the error.

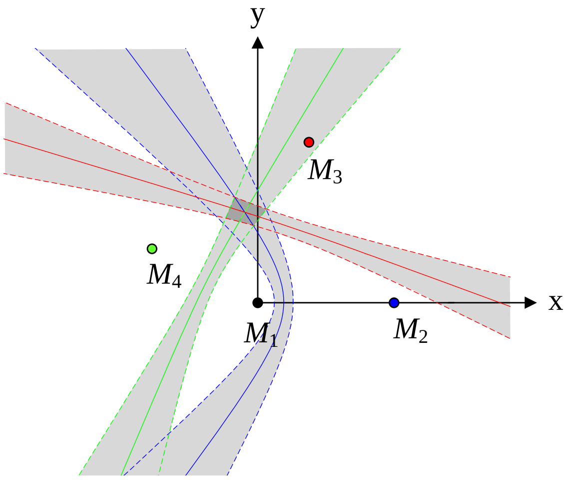

Fig. 1. The measured time difference $\Delta t_{k\ell j}$ between M_k and M_ℓ yields a hyperbola (solid lines). Inequation (4) bounds a region of uncertainty where S_j can reside (grey regions), here depicted for the origin M_1 and three other receivers.

Lemma 1. *The inequality (4) for $k = 1$, $\ell \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$ follows from the $\frac{1}{2}\epsilon$-approximation problem of localization.*

From inequality (4) for $k = 1$, $\ell \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$ follows a solution for the ϵ-approximation problem.

Proof.

"$\Rightarrow$": Using the inequalities (2) and eliminating u_j leads to inequality (4) with 2ϵ.
"$\Leftarrow$": Choose $u_j = t_{1j} - \|M_1 - S_j\|$. Then, the approximation inequality (2) follows for $k > 1$. Note that for $k = 1$ we obtain Equation 1.

The inequality (4) describes an *uncertainty band* for the position of the signal source M_j enclosed by the hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} - \epsilon)$ and $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} + \epsilon)$, see Fig. 1. We compute the intersections of these areas for all M_ℓ for $\ell > 1$. If the intersections of these areas are non-empty, then for these receivers positions a signal source exists. If these areas exist for all signal sources, then there is a solution to the approximation problem. So, the test problem is reduced to a computational geometry problem.

Lemma 2. *The intersection of the $n - 1$ uncertainty bands for a signal source S_j can be computed in time $\mathcal{O}(n^2)$.*

Proof. We use a sweep line technique where the sweep line is a half-line starting at the origin M_1 rotating around it. This is motivated by the observation that such a sweep line intersects each hyperbola $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} \pm \epsilon)$ at most once.

Furthermore in polar coordinates the intersection band is beyond the union of all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} - \epsilon)$ and below all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} + \epsilon)$.

In this proof we compute the curve in polar coordinates which is defined by the farthest points of all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} - \epsilon)$ on the sweep-line with angle α. Then we compute the curve of the closest points of all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} + \epsilon)$ on the sweep-line with angle α. We describe both points by the section-wise definition of the curves. Then, we test in linear time (depending on the number of sections) whether between the $-\epsilon$-curve and the ϵ-curve an area exists.

Curve of farthest points of all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} - \epsilon)$ Each hyperbola is only defined in a sector framed by the central angle interval. So, first we compute the intersection of the angle intervals, which can be done by sorting start and end angles in time $\mathcal{O}(n \log n)$. Note that two hyperbolas intersect in at most two points and that these points can be computed directly. By the definition of the Davenport-Schinzel-sequences [18] the number of intersections in the curve is bounded by $\lambda_2(n-1) = 2n-3$.

We compute the curve by starting with the hyperbola $\mathcal{H}(1, i_0, j, \Delta t_{1 i_0 j} - \epsilon)$ corresponding to the start of the intersected angle interval. Then we compute for this hyperbola all intersections with other hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} - \epsilon)$ and test whether they cross this hyperbola. We take the first (right-handed seen from the origin) such hyperbola following the sweep-line approach. So, we get the next hyperbola corresponding to receiver M_{j_1} in time $\mathcal{O}(n)$. We repeat this search getting receivers $M_{j_2}, M_{j_3}, \ldots$ until we have arrived at the end of the intersecting angle interval. Since the number of intersection is bounded by $2n-3$, the overall run-time is $\mathcal{O}(n^2)$.

Curve of closest points of all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} + \epsilon)$ Now we have to compute the union of all angle intervals of the hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} + \epsilon)$. Again this can be done by sorting the angles in time $\mathcal{O}(n \log n)$. If the curve is not defined over the full range we can use the above method for each disjoint interval. In the other case we need to compute a starting point. For this we compute the nearest points (seen from the origin M_1) for each of the $n-1$ hyperbolas which can be done in closed form. The hyperbola segment around this point is part of the solution. Starting from this hyperbola we can use the analogous algorithm to explore the segments of the curve. Note that in this case the number of intersections is slightly higher, since one additional intersection may occur because of the circular nature of the functions yielding $\lambda_2(n-1)+1 = 2n-2$ hyperbola segments. Again we needs time $\mathcal{O}(n)$ for finding the next segment. So, the overall computation time is again $\mathcal{O}(n^2)$.

Testing the non-emptiness of the intersection Both curves are given as sorted lists of the hyperbolas, according to the angles. Each of them has at most $2n-1$ segments. We join the set of angles and test for each interval on which the hyperbolas lie and whether intersections occur. For each of the intervals this can be done in constant time, resulting in an additional effort of time $\mathcal{O}(n)$.

Lemma 3. *The test of the feasibility of receiver positions can be performed in time $\mathcal{O}(n^2 m)$.*

Proof. This follows from repeating the above test for all m signals.

3.2 Recursive Search for the Receiver Positions

Now, one could enumerate all grid positions of distance $\frac{1}{\sqrt{2}}\epsilon$ and use the test to solve the approximation problem resulting in run-time $\mathcal{O}((\sqrt{2}/\epsilon)^{2n-3} mn^2)$ much alike the trivial algorithm. However, there is a much more efficient method which reduces the time behavior in practical tests. In fact empirical tests point towards an average run-time of $O((-\ln \epsilon)n^2 m)$ for the now following approach.

We consider a recursive tree construction shown in Algorithm 1, where the $2n-3$-dimensional subgrid is repartitioned by a factor of two in each iteration. It uses the feasibility test described in the previous subsection. In the uppermost grid (consisting of only one cell) we choose $\hat{\epsilon} = \sqrt{2}$ and decrease this value in each level by a factor of two. If we have reached $\hat{\epsilon} \leq \epsilon_{\text{target}}$ the search algorithm stops. For each level we discard non-feasible sub-cells. This way, we avoid exhaustive search by pruning the search tree at a higher level. Simulations indicate that in the long run the number of sub-cells remains at most in the hundreds. However, we can show only the run-time of the trivial method which is not very efficient.

Theorem 2. *Algorithm 1 solves the approximation problem in time $\mathcal{O}((\sqrt{2}/\epsilon)^{2n-3} mn^2)$.*

Proof. The theorem follows already from the trivial test algorithm which tests all points in the grid. Note that the improved Algorithm 1 also satisfies this bound.

We prove the correctness of Algorithm 1. Clearly, if the algorithm finds a solution then Lemma 1 implies that it is a solution for the approximation problem. It remains to show that a solution is always found.

Now consider the solution of the localization problem of $\hat{M}_i$. Then, in a grid of cell size s for each receiver $\hat{M}_i$ there is a point M_i within distance $\sqrt{2}s$ with the exception of $\hat{M}_1$ which we define as $M_1 = (0, 0)$. Inequality (4) holds for $\epsilon = \sqrt{2}s$. Therefore this set of receivers would not be discarded in the feasibility test as long as $\epsilon \geq \sqrt{2}s$. Algorithm 1 ensures that sub-cells are only erased from the queue until ϵ is smaller. Therefore, the cell with center M_i is not removed from the queue unless a smaller cell containing the solution is inserted.

4 Polynomial Time Approximation for Small Numbers of Signals

Now, we consider the case where the number of signal sources is small, e.g. $m = 3$. Now, we search possible signal locations and test whether these positions are feasible.

Algorithm 1 Breadth-first search for ϵ-environment for n receivers

Require: Given initial guess of receiver positions $M := \{M_1, \ldots, M_n\}$, receiver times $(t_{i,j})_{i\in[n],j\in[m]}$, target ϵ_{target}

queue $Q \leftarrow \emptyset$
$M \leftarrow (0,0)^n$,
$Q \leftarrow Q.\text{enqueue}\left((0, M)\right)$
repeat
    $(d, M') \leftarrow Q.\text{pop}()$
    for all $b_1, \ldots, b_{2n-3} \in \{-1, 1\}^{2n-3}$ **do**
        $\tilde{M} = M' + 2^{-d-1} \cdot ((0,0), (b_1, 0), (b_2, b_3), \ldots, (b_{2n-2}, b_{2n-3}))$
        if ReceiverPositionIsFeasible$\left(\tilde{M}, (t_{*,*}), 2^{-d-\frac{1}{2}}\right)$ **then**
            $Q \leftarrow Q.\text{enqueue}\left((d+1, \tilde{M})\right)$
        end if
    end for
until $Q = \emptyset$ **or** $2^{-d-\frac{1}{2}} \leq \epsilon_{\text{target}}$
return $Q.\text{pop}()$

4.1 A Test for the Feasibility of Signal Source Positions

We revisit the inequalities (4), but now we consider M_ℓ as a variable while $M_k = M_1$ and S_j is fixed. Given the positions $S_1, \ldots, S_m$, we consider all possible locations for a receiver M_i. These are intersections of disks and complements of disks. If the intersections are non-empty, then the signal locations are feasible.

Lemma 4. *The intersection of the m disks and m disk complements can be computed in time $O(m \log m)$.*

Proof. The intersection of m disks can be computed in time $\mathcal{O}(m \log m)$ [2]. The union of m disks can be also computed in time $\mathcal{O}(m \log m)$ [2], which describes the intersection of the disk complements. Using a sweep line technique one can now test whether the intersection of these two areas is empty. The time for this is also bound by $\mathcal{O}(m \log m)$.

Given the boundaries of the intersections a sweep line method can compute the intersecting area. The sweep line events occur when the sweep line hits a segment of the two boundary paths the first time, when segments intersect, or when the sweep line leaves a segment. All these events can be computed in time $\mathcal{O}(m \log m)$ in advance or (for the intersection of the two boundaries) in time $\mathcal{O}(\log m)$ when a new boundary segment is introduced. Applying the sweep line method computes the wanted intersection in time $\mathcal{O}(m \log m)$, since at most $\mathcal{O}(m)$ events take place.

Lemma 5. *The test for the feasibility of signal positions can be performed in time $O(nm \log m)$.*

Proof. This follows from repeating the test for all $n - 1$ receivers.

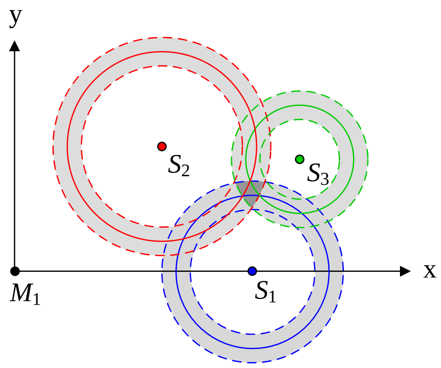

Fig. 2. Given a time difference Δt between a fixed receiver M_1 and an unknown receiver M_ℓ. Now, a fixed location for S_1, S_2, and S_3 is assumed. The hyperbolic inequality (4) is satisfied if the unknown receiver M_ℓ resides in a circular epsilon environment (grey regions).

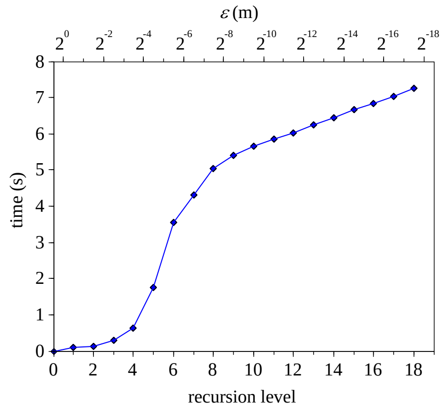

Fig. 3. Total run-time of an example with four receivers and 20 signals on an Intel Core-i5 quad-core CPU to achieve precision ϵ. The recursion level is $\delta = 0.5 - \log_2(\epsilon)$. The run-time is proportional to the number of processed nodes.

4.2 Recursive Search for the Signal Positions

This search is completely analogous to Section 3.2 and Algorithm 1 while we now search for signal positions.

Theorem 3. *The analogous search approximation algorithm solves the problem in time* $\mathcal{O}(\left(\sqrt{2}/\epsilon\right)^{2m-3} nm \log m)$.

Proof. The trivial grid oriented algorithm testing all positions of in a grid of cell size $\epsilon/\sqrt{2}$ combined with the feasibility test above yields the run-time.

Again a tree-based search performs better for real-world inputs, but does not give better worst-case bounds.

5 Empirical Results

We concentrate on the case of four receivers and a large number of signal sources, since the practical impact of this problem is higher. We have implemented our algorithm in a computer algebra system as well as in C++. As the inner nodes of the search tree are independent the problem is inherently parallel. We profit from this characteristic and execute multiple threads on modern multi-core CPUs.

For our experiments we generate the positions of four receivers randomly in the unit circle. They are transformed such that the first location is in the origin, the second is on the positive leg of the x-axis, and the third receiver is in the upper two quadrants.

Now m signals are sampled at random positions in the unit circle. For m we choose a series from 5 to 50 signals with 100 runs for each number of signals. The time differences of arrival are calculated and passed to the algorithm, which is the only information given to the algorithm.

We have evaluated the algorithm in terms of inspected search nodes, search queue length, duration and correctness of the calculation, i.e. the receiver positions.

In some under-determined cases with few signal sources, or malicious receiver positions, where any two receivers are very close to each other, we observe run-times as indicted by the worst case analysis. Then, the width of the search tree grows rapidly resulting in high run-times. In these cases we abort the algorithm after 10 minutes and mark the attempt failed (although the algorithm would eventually find the solution).

The number of required evaluation steps depends on the location characteristics of the receivers, on the traversal strategy through the tree, and on the number of signals. In our simulations we choose breadth-first search, which is the slowest search type, but with deterministic characteristics. Then, given a sufficient number of signals, the number of traversed nodes varies between 10^4 and 10^7 with a cumulation at 10^5. With decreasing number of signals the algorithm has increased difficulty to eliminate possible locations, increasing the number of steps by a factor of 10^1 and more.

On an Intel Core-i5 machine we could process a number of 10^5 nodes in about $4-8$ seconds, running on four processor cores (Fig. 3). A typical execution time given 40 signals is 8 seconds, which is the mode of the distribution of runtimes, with some ill-conditioned settings raising the mean to 10.5 seconds. Alltogether, the typical execution time is 10 seconds with a mean of 44 seconds.

At the current stage our implementation is too slow for real-time applications on standard PC hardware. However the prospects are great, as the search algorithm can be so easily run in parallel. On typical computers, and even smartphones, the number of processing cores increases permanently and our algorithm can benefit with efficiency: Our search algorithm does speed up with both higher core performance and higher number of cores, other than iterative algorithms which usually profit only from increases in per-core performance.

6 Conclusions and Future Work

In this contribution we have presented what is, to our knowledge, the first solution for the TDOA-Self-Localization problem given the minimum number of four receivers. In our model the uncertainties of TDOA measures are expressed by an ϵ-approximation scheme, returning the best explanation of receiver locations for the given time-of-arrival data of signals from unknown locations. If the data is precise up to an error of ϵ then also the receiver locations are determined up to an error in the order of ϵ.

For refining the estimation of our ϵ-model we use a fully polynomial time approximation scheme running in time $\mathcal{O}((\sqrt{2}/\epsilon)^{2n-3}n^2m)$ for the receiver problem and in $\mathcal{O}((\sqrt{2}/\epsilon)^{2m-3}nm\log m)$ for the analogous problem of estimating small numbers of at least three signals. This implies the following corollary.

Corollary 1. *The approximation of the four receiver localization problem can be solved in time $\mathcal{O}(\epsilon^{-5}m)$ for m signal sources. The problem of the three signal localization can be solved in time $\mathcal{O}(\epsilon^{-5}n)$ for n receivers.*

We have implemented the algorithm in a multi-threaded simulation of randomized receiver locations in the unit disc. We could show the feasibility of our approach and we could show that we traverse the search tree in a couple of seconds in most cases.

In some cases our approach suffers from an ill-conditioned configuration of the receiver locations, i.e. some of the four receivers are close to each other or near to a line, rendering the problem close to under-determined. Then, our algorithm is forced to generate a very large search tree, resulting in a long duration for the traversal. However, when we find a solution we are guaranteed that it is correct, up to the order of ϵ.

One open problem is whether the intersection of n halfspaces bordered by hyperbolas can be computed in time $\mathcal{O}(n\log n)$ like the intersection of disks. To our knowledge nobody has addressed this non-trivial problem so far.

We have seen that our algorithm suffers from a large search tree in some cases, and according to that long execution times. There are worst-case inputs where this is inevitable, i.e. if the receivers are located within a radius of ϵ. But also for non-degenerated inputs we see plenty of room for improvements, as we use Breadth-First-Search for now to traverse the tree in a deterministic manner, which allows to draw conclusion about the expected run-time.

If we use Depth-First-Search we can use heuristics to choose branches of the search tree. The task is to develop such a model using elementary approximation schemes as for example presented in [6] and [17].

Acknowledgements This work has partly been supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) within the Research Training Group 1103 (Embedded Microsystems). We thank Wolfram Burgard for the discussions that inspired us to come up with this solution.

References

1. Biswas, R., Thrun, S.: A Passive Approach to Sensor Network Localization. In: Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004). vol. 2, pp. 1544–1549 (2004)
2. Brown, K.Q.: Geometric Transforms for Fast Geometric Algorithms. Ph.D. thesis, Carnegie-Mellon University Pittsburgh PA, Dept. of Computer Science (1979)
3. Dakin, R.J.: A tree-search algorithm for mixed integer programming problems. The Computer Journal 8(3), 250–255 (1965)
4. Gillette, M.D., Silverman, H.F.: A Linear Closed-Form Algorithm for Source Localization From Time-Differences of Arrival. Signal Processing Letters, IEEE 15, 1–4 (2008)

5. Hongyang, C., Ping, D., Yongjun, X., Xiaowei, L.: A Robust Location Algorithm with Biased Extended Kalman Filtering of TDOA Data for Wireless Sensor Networks. In: Proceedings of the 2005 International Conference on Wireless Communications, Networking and Mobile Computing. vol. 2, pp. 883–886. IEEE (2005)
6. Janson, T., Schindelhauer, C., Wendeberg, J.: Self-Localization Application for iPhone using only Ambient Sound Signals. In: Proceedings of the 2010 International Conference on Indoor Positioning and Indoor Navigation (IPIN). pp. 259–268 (Nov 2010)
7. Janson, T., Schindelhauer, C., Wendeberg, J.: Self-localization Based on Ambient Signals. In: Algorithms for Sensor Systems, Lecture Notes in Computer Science, vol. 6451, pp. 176–188. Springer (2010)
8. Kennedy, D.: Some branch and bound techniques for nonlinear optimization. Mathematical Programming 42(1), 147–157 (1988)
9. Land, A.H., Doig, A.G.: An Automatic Method of Solving Discrete Programming Problems. Econometrica 28(3), 497–520 (1960)
10. Lawler, E.L., Wood, D.E.: Branch-And-Bound Methods: A Survey. Operations Research 14(4), pp. 699–719 (1966)
11. Moses, R.L., Krishnamurthy, D., Patterson, R.M.: A Self-Localization Method for Wireless Sensor Networks. EURASIP Journal on Advances in Signal Processing pp. 348–358 (2003)
12. Pollefeys, M., Nister, D.: Direct computation of sound and microphone locations from time-difference-of-arrival data. In: IEEE International Conference on Acoustics, Speech and Signal Processing. pp. 2445–2448. IEEE (2008)
13. Priyantha, N.B., Chakraborty, A., Balakrishnan, H.: The Cricket Location-Support System. In: MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking. pp. 32–43 (2000)
14. Raykar, V., Duraiswami, R.: Automatic position calibration of multiple microphones. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'04). vol. 4, pp. iv–69. IEEE (2004)
15. Rui, Y., Florencio, D.: New direct approaches to robust sound source localization. In: Proceedings of IEEE ICME 2003. pp. 6–9. IEEE (2003)
16. Savvides, A., Han, C.C., Strivastava, M.B.: Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors. In: Proceedings of the 7th annual international conference on Mobile computing and networking. pp. 166–179. ACM (2001)
17. Schindelhauer, C., Lotker, Z., Wendeberg, J.: Network Synchronization and Localization Based on Stolen Signals. In: Proceedings of 18th International Colloquium on Structural Information and Communication Complexity (SIROCCO) (2011)
18. Sharir, M., Agarwal, P.K.: Davenport Schinzel sequences and their geometric applications. Cambridge Univ. Press, New York (1995)
19. Stewénius, H.: Gröbner Basis Methods for Minimal Problems in Computer Vision. Ph.D. thesis, Lund University (Apr 2005)
20. Thrun, S.: Affine Structure From Sound. In: Proceedings of Conference on Neural Information Processing Systems (NIPS). MIT Press, Cambridge, MA (2005)
21. Valin, J.M., Michaud, F., Rouat, J., Létourneau, D.: Robust Sound Source Localization Using a Microphone Array on a Mobile Robot. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS). pp. 1228–1233 (2003)
22. Wendeberg, J., Höflinger, F., Schindelhauer, C., Reindl, L.: Anchor-free TDOA Self-Localization. In: Proceedings of 2011 International Conference on Indoor Positioning and Indoor Navigation (IPIN) (2011)