Comparison of Positioning Algorithms for a RTOF Radar System based on Multipath Simulations

Silvan Wehrli and Heinz Jäckel
Electronics Laboratory, ETH Zurich, 8092 Zurich, Switzerland. Email: silvan.wehrli@alumni.ethz.ch

***Abstract*—In round-trip time-of-flight (RTOF) based radar systems, the position of the mobile station can be calculated by trilateration. However, it is possible that the spheres do not intersect due to noisy distance measurements. Hence, a different localization algorithm has to be applied. This paper compares the performance of a least-squares algorithm, a linear optimization algorithm and mass-spring models based on multipath simulation data. The mass-spring model has a shorter computation time as the least-squares algorithm ($\sim$10 ms vs. $\sim$32 ms) and is as accurate as the least-squares algorithm.**

The number of available base stations and the placement of the base stations strongly influences the positioning accuracy. The simulated distances with multipath propagation have an accuracy of 17.2 cm. The 3D accuracy with 8 base stations (σ_x=8.9 cm, σ_y=9.9 cm, σ_z=14.5 cm) is better than the individual distance measurements.

***Index Terms*—Radar, RToF, positioning algorithms, indoor positioning, multipath propagation.**

I. Introduction

In recent years, many new indoor positioning systems with differential measurement principle emerged. A promising approach is to use RF signals, because of the larger range compared to ultrasound systems and the circuits can be integrated, which reduces the cost of such a system. The main challenge of indoor positioning with RF signals is strongly overlapping multipath propagation.

One example of such a system is an FMCW radar system with a switched injection-locked oscillator [1], [2], [3] working as an active pulsed reflector (APR) [4]. The system consists of several base stations (BS) and one or more reflectors (mobile stations (MS)). Each base station subsequently measures the round-trip time-of-flight (RTOF) to the reflector. A server collects all the measured distances and calculates the position of the reflector accordingly. This paper evaluates different positioning algorithms based on multipath simulation results.

The three main criteria for a good positioning algorithm for our system are the computation time, the accuracy and the algorithm size. A short computation time enables real-time capability of the system. The computation time should be in the same order of magnitude as the measurement time. For our system, one measurement with 8 base stations and 1 reflector takes below 8 ms. At the same time, the algorithm should give accurate results and it should be simple enough, so that it can be implemented on an FPGA or a small, power-efficient server.

II. Positioning algorithms

It is assumed that all the positions of the base stations $\vec{x}_{BS,i}$ are known a priori, and each base station i measures the distance to the reflector as ρ_i. By knowing the distance to a base station, the possible reflector positions lie on a sphere with the base station at the center. The intersection of the spheres of all base stations gives the correct position of the reflector. This triangulation approach only works, if all the distance measurements are correct. Otherwise, the spheres do not intersect at a specific point. Thus the approach with the intersection of spheres is not an optimal solution in an environment with multipath propagation.

A. Least-Squares

The simplest solution to estimate the position of the reflector $\vec{x}_{ref,det}$, based on the measurement data, is by using an unweighted minimum least-squares algorithm. The least-squares algorithm minimizes the sum of the quadratic error between the measured distances ρ_i and the distances between the base stations i and a possible unknown reflector position $\vec{x}_{ref}$. The least-squares problem can be written as:

$$\vec{x}_{ref,det} = \underset{\vec{x}_{ref}}{\arg\min} \sum_i \left(||\vec{x}_{ref} - \vec{x}_{BS,i}|| - \rho_i\right)^2 . \quad (1)$$

where $\vec{x}_{ref}$ are all possible positions of the reflector. This optimization problem can be solved numerically with the aid of Matlab's optimization toolbox. The interior trust region approach [5], [6] is selected to solve the problem, because this approach is orders of magnitude faster then the brute force grid search approach.

B. Linear Minimization

The least-squares algorithm is not optimal, if all measurements ρ_i with the exception of one are correct. Then, the wrong measurement influences the position strongly, because least-squares tries to balance the errors. The linear minimization can reduce this problem. Linear minimization is almost identical to least-squares, but optimizes the 1 norm of all measured distances versus the calculated distances at the point $\vec{x}_{ref}$.

$$\vec{x}_{ref,det} = \underset{\vec{x}_{ref}}{\arg\min} \sum_i \left|\left(||\vec{x}_{ref} - \vec{x}_{BS,i}|| - \rho_i\right)\right| . \quad (2)$$

One single distance measurement error has a smaller weight on the sum (not squared), and thus influences the detected position less. It could be that often only one distance measurement

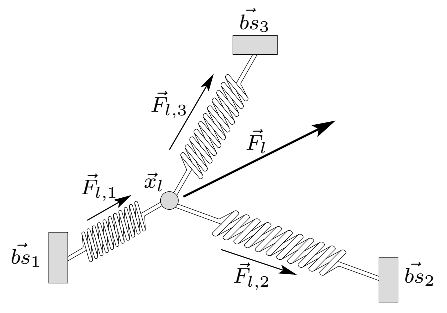


Fig. 1. Illustration of the mass-spring model for one mobile reflector $\vec{x}$ and three base stations $\vec{bs}_1$, $\vec{bs}_2$ and $\vec{bs}_3$.

is wrong in a multipath environment. Hence, we also test linear minimization.

C. Mass-Spring Model

A popular algorithm used to solve the localization problem in sensor networks is the mass-spring model. It was originally presented by [7] for modeling the drape of clothes and improved by [8]. A cloth object is approximated by a deformable surface composed of masses and springs, the movement of which is evaluated using the numerical integration of the fundamental law of dynamics.

A wireless sensor network can also be viewed as a mesh. The sensors measure the distances to the closest neighbors, which is similar to the connection of two points with a thread. Thus, the same algorithms can be applied [9]. But in contrast to clothes only the final state is of interest in sensor networks. Therefore, a simplified algorithm can be used [10]. The situation with fixed base stations and one movable reflector, as it is the case in our positioning system, is a very simple sensor network. Thus, the mass-spring algorithm can be applied.

Figure 1 illustrates the mass-spring model for three base stations and one reflector. The algorithm starts by assuming a position of the reflector $\vec{x}_1$. Usually the middle of the room is chosen, but this depends on the application. In a next step, the distances between the initial position $\vec{x}_1$ (or more generally $\vec{x}_l$ for the lth iteration) and the base stations $\vec{bs}_i$ are calculated by

$$\vec{d}_{l,i} = \vec{bs}_i - \vec{x}_l, \tag{3}$$

which can be split up in a direction vector

$$\vec{d}_{l,i,dir} = \frac{\vec{d}_{l,i}}{||\vec{d}_{l,i}||} \tag{4}$$

and a length

$$d_{l,i} = ||\vec{d}_{l,i}||. \tag{5}$$

A virtual spring is inserted between each base station and the reflector. Each spring i exerts a force $\vec{F}_{l,i}$ on the reflector depending on the difference between the measured distance ρ_i and the estimated distance $d_{l,i}$ between the ith base station and the estimated position $\vec{x}_l$

$$\vec{F}_{l,i} = k\left(d_{l,i} - \rho_i\right)\vec{d}_{l,i,dir} \tag{6}$$

where k is the linear spring constant. The total force exerted on the reflector is

$$\vec{F}_l = \sum_i \vec{F}_{l,i}. \tag{7}$$

The complete mass-spring model calculates the complete dynamics of the system. The mobile station has a certain mass m and a viscous damping factor C_{vis}. The total force $F_{tot,l}$ in iteration l is the sum of the spring forces and the damping

$$\vec{F}_{tot,l} = \vec{F}_l - C_{vis}\vec{v}_l \tag{8}$$

This force $\vec{F}_{tot,l}$ accelerates the mobile station to a new speed $\vec{v}_{l+1}$

$$\vec{v}_{l+1} = \vec{v}_l + \frac{\vec{F}_{tot,l}}{m}\Delta t \tag{9}$$

where Δt is the incremental time step. The new estimated position $\vec{x}_{l+1}$ is now calculated by

$$\vec{x}_{l+1} = \vec{x}_l + \vec{v}_l\Delta t \tag{10}$$

The algorithm continues with the calculation of the distances $\vec{d}_{l+1,i}$ based on the new estimated position $\vec{x}_{l+1}$. The algorithm stops, when the resulting force F_l is below a certain limit or the estimated position does not change anymore. The parameters m, C_{vis}, k and Δt can be freely chosen. The system can be overdamped, critical damped, under damped or even unstable. The optimal parameter set has to be found by testing different scenarios.

We are only interested in the final position, where the force vector $\vec{F}_l$ is minimal. In the mass-spring model, the mass (mobile station) is accelerated into the direction of the resulting force, damped by the viscosity factor and moves with a certain speed. Instead of calculating the whole dynamics of the mass-spring model with acceleration and viscosity, the point with the lowest potential energy can be found with a gradient descent. We assume that the mobile station moves in the direction of $\vec{F}_l$ with a proportional factor k_2 [s^2/kg]

$$\vec{x}_{l+1} = \vec{x}_l + k_2\vec{F}_l, \tag{11}$$

The speed $\vec{v}_l$ and the damping are neglected. Hence, fewer calculation steps are needed for each iteration. The number of parameters is reduced to one ($k_{tot} = k \cdot k_2$). The larger k_{tot} is, the faster the algorithm converges until the algorithm gets unstable. We experimentally verified that the algorithm is stable for $k_{tot} \leq 0.3$. A smaller value $k_{tot} = 0.1$ ensures a stable algorithm. The advantages of the mass-spring model are the simple implementation and the fast convergence.

This algorithm is a form of gradient descent, thus it cannot be assured that the found minimum is the global minimum. A typical plot of the strength of the force $\vec{F}_l$ for a 2D localization problem with 3 base stations (at $\vec{x}_{bs1} = [x, y] = [0, 0]$, $\vec{x}_{bs2} = [0, 10]$ and $\vec{x}_{bs3} = [10, 10]$) is depicted in Fig. 2. In this case, only one minimum exists. The red line illustrates the solution of the mass-spring algorithm with a starting point at $\vec{x}_1 = [8, 9]$.

In this paragraph, it will be verified that the mass-spring model is a simple method to solve the least-squares problem.

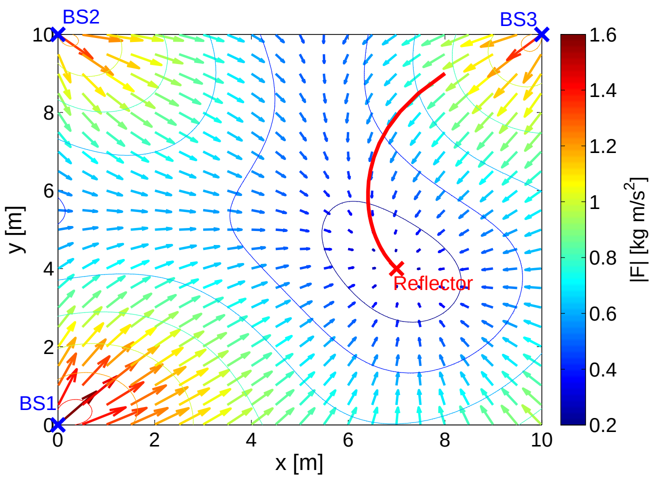


Fig. 2. The resulting force vector ($\vec{F}$) field for 3 base stations at $\vec{x}_{bs1} = [x, y] = [0, 0]$, $\vec{x}_{bs2} = [0, 10]$ and $\vec{x}_{bs3} = [10, 10]$. The actual position of the reflector is at $\vec{x}_{reflector} = [7, 4]$. The red path illustrates the solution of the mass-spring algorithm for a starting point $\vec{x}_1 = [8, 9]$.

The mass-spring optimization minimizes the resulting force on the reflector. By minimizing the force, the algorithm also minimizes the potential energy. The potential energy of one spring i during the lth iteration can be written as

$$E_{pot,l,i} = -\int \vec{F_{l,i}} \mathrm{d}\vec{x} = \frac{1}{2} k \left(d_{l,i} - \rho_i\right)^2 \tag{12}$$

The total potential energy in iteration l is

$$E_{pot,l} = \sum_i E_{pot,l,i} = \frac{1}{2} k \sum_i \left(d_{l,i} - \rho_i\right)^2 . \tag{13}$$

In each iteration, the force vector $\vec{F_l}$ decreases, which results in a lower potential energy. Thus, the minimization of the potential energy solves the least-squares problem (1). The advantages of the mass-spring model are the fast convergence and its simplicity. The mass-spring algorithm can be implemented on an FPGA. The disadvantage is that the algorithm can converge to a local minimum or not at all.

III. Performance Evaluation of Positioning Algorithms

The previously described four positioning algorithms are implemented in Matlab and compared. The least squares and the linear algorithm use the Matlab functions 'lsqnonlin' and 'fmincon' from the 'Optimization Toolbox'. Two versions of the mass-spring algorithm are programmed: the simple one calculates the position according to (11). The complete mass-spring algorithm (ms_c) calculates the complete dynamics of the mass-spring model with Equations (8), (9) and (10). The complete algorithm has the advantage that in some cases, the algorithm will not stop in a local minimum because of the remaining speed. The solution of the complete mass-spring algorithm is comparable to the movement of a marble in the potential well. If only the force is taken into account, the marble can stop in a small local minimum. The complete mass-spring model calculates the dynamics, thus the marble in the local minimum can have enough kinematic energy to get over the potential barrier into the global minimum.

The algorithm should be tested on realistic data. For the system simulation with multipath propagation, a simple scenario was chosen where the multipath components are calculated based on an empty room with length and width of 10 m each and height of 3 m. A similar environment was presented in [11]. For each base station (17 base stations) and reflector position (200), the propagation channel is calculated and fed into the system simulation. The complete positioning system is simulated with Simulink. The final available data are the detected distances ρ_i between each base station and each reflector position in the case of multipath propagation. The standard deviation of the distance error ($\sigma_{\Delta d}$) is 17.19 cm.

TABLE I
Computation time in [ms] for the various positioning algorithms.

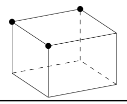
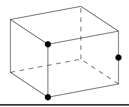
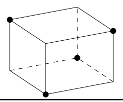
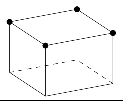
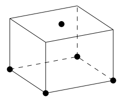
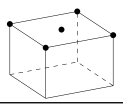
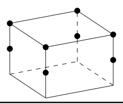
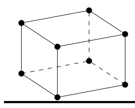
# of bs	t_{lsq}[1]	t_{ms}[2]	t_{ms_c}[3]	t_{lin}[4]
3	31.59	19.50	73.75	71.35
4	32.20	10.71	29.04	75.28
5	33.59	10.55	26.61	78.10
8	32.36	9.18	20.58	74.93

[1] least-squares, [2] mass-spring, [3] complete mass-spring, [4] linear algorithm

A. Computation Time

Table III-A summarizes the average computation times of these algorithms. The computation time is averaged over 200 different reflector positions and then 4 different sets of base stations. This leads to a total of 800 executions of each algorithm for a certain number of base stations. The least-square and the linear algorithm are independent of the number of base stations. Both mass-spring algorithms profit from an increasing number of base stations. For 3 base stations, the 3D position is the intersection of three spheres. The intersection of two spheres is a circle or a point (if they intersect). The intersection of the circle with a sphere results in two possible solutions. Hence, the solution is ambiguous. The algorithm can detect the correct position in most cases, because only one of the two solutions is inside the room. However, the force distribution is very flat between the two possible solutions and thus the mass-spring algorithms converge slower. Nevertheless, the simple mass-spring algorithm outperforms all others. For a higher number of base stations, the force which drags the MS to the correct position increases and thus the mass-spring algorithms converge faster. For more than 3 base stations, the complete mass-spring model is the second fastest. The mass-spring model is solved in 9.18 ms for 8 base stations. This is in the same order as the time needed for the 8 base stations to measure the distance to the reflector. Thus, the system is as fast in calculating the position as it is measuring the single distances, which is a prerequisite for real-time application.

TABLE II
COMPARISON OF THE ACCURACY OF THE 4 IMPLEMENTED POSITIONING ALGORITHMS. ALL VALUES ARE IN [CM]. SIMULATED ROOM IS 10 M WIDE, 10 M LONG AND 3 M HIGH.

		lsq	ms	ms_c	lin
	σ_x	18.666	18.964	18.966	18.375
	σ_y	17.955	18.253	18.256	18.253
	σ_z	69.596	66.935	66.956	68.297
	σ_x	18.888	18.902	19.226	20.135
	σ_y	27.684	29.812	27.345	29.197
	σ_z	52.853	52.549	60.437	52.899
	σ_x	14.808	14.889	14.888	16.982
	σ_y	14.089	13.201	13.202	15.819
	σ_z	37.514	31.585	31.584	35.059
	σ_x	12.504	12.486	12.486	14.786
	σ_y	12.756	12.754	12.755	14.864
	σ_z	36.718	35.485	35.482	41.301
	σ_x	11.525	11.389	11.389	13.846
	σ_y	12.290	12.307	12.307	14.072
	σ_z	23.133	23.092	23.096	27.314
	σ_x	11.230	11.043	11.043	13.281
	σ_y	12.274	12.262	12.262	14.193
	σ_z	32.058	31.193	31.193	44.304
	σ_x	9.570	9.559	9.560	11.638
	σ_y	9.938	9.950	9.950	11.861
	σ_z	29.607	29.573	29.580	34.568
	σ_x	8.921	8.916	8.916	11.650
	σ_y	9.861	9.861	9.861	12.037
	σ_z	14.573	14.540	14.539	19.581

B. Accuracy

Table II compares the accuracy of these algorithms for different base station subsets. The accuracy is the standard deviation of the position error $\Delta\vec{x} = \vec{x}_{ref,det} - \vec{x}_{reflector}$. Overall, the linear minimization algorithm results in the highest standard deviation and thus is the worst algorithm. However, the difference between the linear algorithm and the other ones is smaller then in the 2D case. The simple (ms) and the complete (ms_c) mass-spring model perform identically for all except one case. The difference between the least-square algorithm and the simple mass-spring model is also marginal. Thus, we can conclude that the simple mass-spring model combines the fastest computation time and one of the best accuracies, which makes this algorithm suitable for real-time indoor positioning.

Table II also indicates on where the best placements for the base stations are. Only a few of the evaluated base stations subsets are listed. Configurations, where 2 solutions for the same measurement are possible, are neglected. For example this occurs, when all base stations are 1.5 m above the floor, because for every reflector's z-component two solutions exist ($1.5\ m \pm z_{component}$). In Table II, always the best configurations for a certain number of base stations are depicted together with some other configurations. It is obvious that more base stations result in a more accurate position, because the individual distance errors have a smaller weight on the final position. The 2D accuracy (σ_x and σ_y) outperforms the distance measurement accuracy ($\sigma_{\Delta d} = 17.19$ cm), if there are 4 or more base stations. The z-component improves when the base stations are in different heights.

IV. CONCLUSION

The comparison of the four positioning algorithms (linear, least-squares, mass-spring model and complete mass-spring model) based on multipath simulation data revealed that the mass-spring model is the fastest. Furthermore, the mass-spring model converges faster, if more base stations are available. The linear and the least-squares algorithms do not scale with increasing number of base stations. The accuracies of the mass-spring model and the least-squares solution are almost identical. Thus, the mass-spring model is a effective algorithm for the position calculation in a RTOF based positioning system even with multipath errors.

REFERENCES

[1] M. Vossiek, L. Wiebking, P. Gulden, J. Wieghardt, C. Hoffmann, and P. Heide, "Wireless local positioning," *Microwave Magazine, IEEE*, vol. 4, pp. 77 – 86, Dec. 2003. I

[2] L. Wiebking, *Entwicklung eines zentimetergenauen mehrdimensionalen Nahbereichs-Navigations-Systems*. VDI Verlag, 2003. I

[3] M. Vossiek and P. Gulden, "The switched injection-locked oscillator: a novel versatile concept for wireless transponder and localization systems," *IEEE Trans. Microwave Theory Tech.*, vol. 56, pp. 859–866, apr 2008. I

[4] S. Wehrli, R. Gierlich, J. Huttner, D. Barras, F. Ellinger, and H. Jackel, "Integrated active pulsed reflector for an indoor local positioning system," *Microwave Theory and Techniques, IEEE Transactions on*, vol. 58, pp. 267 –276, Feb. 2010. I

[5] T. F. Coleman and Y. Li, "An interior trust region approach for nonlinear minimization subject to bounds," *SIAM Journal on Optimization*, vol. 6, no. 2, pp. 418–445, 1996. II-A

[6] T. F. Coleman and Y. Li, "On the convergence of interior-reflective Newton methods for nonlinear minimization subject to bounds," *Math. Program.*, vol. 67, no. 2, pp. 189–224, 1994. II-A

[7] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, "Elastically deformable models," in *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, vol. 21, (New York, NY, USA), pp. 205–214, ACM Press, July 1987. II-C

[8] X. Provot, "Deformation constraints in a mass-spring model to describe rigid cloth behavior," in *Graphics Interface*, pp. 147–154, 1995. II-C

[9] W. Chen, T. Mei, M. Q.-H. Meng, H. Liang, Y. Liu, Y. Li, and S. Li, "Localization algorithm based on a spring model (LASM) for large scale wireless sensor networks," *Sensors*, vol. 8, no. 3, pp. 1797–1818, 2008. II-C

[10] A. Howard, M. Mataric, and G. Sukhatme, "Relaxation on a mesh: a formalism for generalized localization," in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, vol. 2, pp. 1055–1060 vol.2, 2001. II-C

[11] S. Wehrli and H. Jäckel, "Non-stochastic multipath simulations for an indoor local positioning system," in *2010 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Sep. 2010. III