An Indoor Localization Framework for Four-Rotor Flying Robots Using Low-Power Sensor Nodes

Juergen Eckert, Reinhard German, and Falko Dressler, *Senior Member, IEEE*

***Abstract*—We study the relevance of highly precise indoor localization techniques for quadrocopters and present an ultrasonic sensor system that achieves excellent localization performance, although the system has to rely on the limited computation resources of sensor nodes used for time-of-flight-based localization. Quadrocopters, i.e., flying four-rotor robots, are on-board sensor controlled systems. In comparison to classical monorotor objects (helicopters), the quadrocopters can be piloted with much lower effort. However, lateral drifts cannot be compensated for only referring to the built-in sensors. Nonetheless, the detection of such drifts is strongly necessary for indoor operation—without any corrections, the quadrocopter would quickly cause a collision. To compensate for the dislocation, we developed an indoor-localization framework for time-of-flight-based localization using ultrasonic sensors. It is optimized for use in sensor nodes with low computational power and limited memory. The system is designed for high scalability and to provide high accuracy, even in the case of erroneous measurements. The developed hardware platform is very lightweight to be carried by mobile robots and flying quadrocopters. Based on our real-time localization system, position controller and navigation functionality can be implemented.**

***Index Terms*—Flying robot, indoor localization, sensor network, ultrasonic.**

I. Introduction

FLYING four-rotor robots are similar to helicopters. In contrast to mono-rotor systems, these so-called quadrocopters usually have more sensors and more robust controllers available. Helicopters without any sensors or controllers can remotely be controlled by a person. However, due to their physical instability, this condition is not possible for quadrocopters: the system needs to continuously be stabilized. Usually, a combination of gyrometers and acceleration sensors is used to determine its current state. Based on these measurements, a digital controller continuously adjusts the orientation of the platform. Using such controllers, those devices can easily be controlled and piloted by other digital systems such as distributed mobile sensor nodes. Unfortunately, the current position cannot be obtained by only controlling the pitch and the roll angles. The quadrocopter always hovers on top of an air cushion, i.e., flying at low speed of less than 2 m/s. Thus, any minimal measurement error or any airflow may cause a drift to a random direction. The system remains highly instable with respect to position maintenance. Angle corrections must continually be applied, and more than one board instruments needs to be used to keep the flying robot in position.

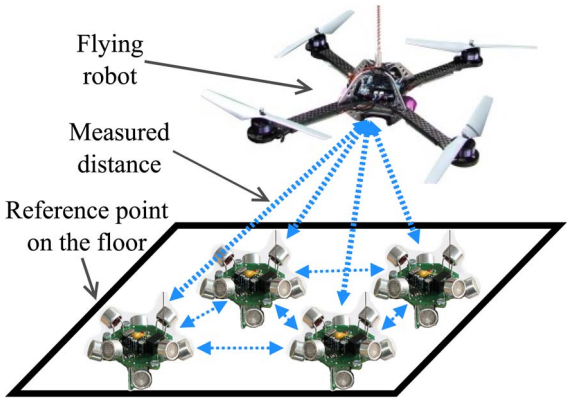

Fig. 1. Four-rotor flying robot hovering over distributed reference points.

Fig. 1 depicts our reference scenario. A quadrocopter relies on an external positioning system to continuously update its system parameters. In general, there are many cases in which applications benefit from more accurate positioning information. Examples include the controlled physical movement of a robot [1] and the location-based information forwarding through an ad hoc network [2].

A discussion of preferences for systems using active and passive mobile devices can be found in [3]. If privacy is an issue, passive localization systems should be preferred. For example, the infrastructure of the Cricket system [4] has no knowledge about the current position of any mobile device. However, this system architecture also has several disadvantages. The accuracy suffers if the mobile device moves during a series of (at least three) measurements. In some cases, e.g., using ultrasound, this limitation is strong, because a set of measurements can take up to several hundred milliseconds. In our scenario, we target to localize a flying quadrocopter, which makes a complete stop during a set of measurements impossible. The object will always drift in a random direction. In active systems, the mobile device emits a signal, and the infrastructure simultaneously receives it. Thus, better accuracies and higher velocities for the mobile devices are possible.

There are a number of localization systems described in the literature, which are based on different measurement and localization techniques. Each of those systems has its benefits and open issues. To the best of our knowledge, no system (neither commercial nor academic) fulfills all the requirements

Manuscript received December 6, 2009; revised April 12, 2010; accepted June 8, 2010. Date of publication October 28, 2010; date of current version January 7, 2011. The Associate Editor coordinating the review process for this paper was Dr. Pierre Payeur.

The authors are with the Department of Computer Science, University of Erlangen–Nuremberg, 91058 Erlangen, Germany (e-mail: juergen.eckert@informatik.uni-erlangen.de; german@informatik.uni-erlangen.de; dressler@informatik.uni-erlangen.de).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIM.2010.2085850

for localizing flying quadrocopters. Real-time localization is frequently an issue; for example, some systems rely on iterative position estimation. Furthermore, many other systems are simply very heavy to be carried by the flying robot. Therefore, we investigated appropriate real-time localization techniques and came up with a new solution that perfectly meets the needs in this application domain [5]. We also implemented this system in our laboratory based on ultrasonic distance measurements.

The main contributions of this paper can be summarized as follows. Primarily, we developed a mathematical procedure and an algorithm for iterative self localization based on time-of-flight (TOF) measurements. The main objective is to enable real-time localization support for distributed sensor networks, which usually have tremendous restrictions in terms of processing power and energy. We aim at using those mobile robots with mounted sensing hardware as reference points for indoor localization that can even be used in unknown territories. The target scenario is our ultrasound sensor system, which is lightweight to be carried by a quadrocopter and by mobile robot systems [6]. The initial work on the indoor localization is presented in [5]. In summary, we provide a framework not only for our chosen scenario but also for other cases of real-time indoor localization.

The rest of this paper is organized as follows. In Section II, we discuss relevant related work. In Section III, we present the mathematical background of our localization system. Then, in Section IV, we discuss some performance aspects of our system. The test system is described in Section V, and relevant communication aspects are discussed in Section VI. Finally, Section VII concludes this paper.

II. Related Work

In the following discussion, we briefly outline some basic principles of localization techniques and approaches related to our work. Hightower and Borriello [7] provide more detailed information on the taxonomy and basic principles of localization techniques.

The basic fundamentals of geometry can be used to compute unknown positions. One possibility is to use the *lateration* technique based on a set of distances. These distances are measured between some reference points, which have known positions, and the object to be localized. Aside from the rather impractical method of *direct* measurements (e.g., with a yard stick), typically, two methods are used. *Attenuation* relies on the fact that an emitted signal decreases in strength during its way. The attenuation is proportional to the distance—in free space, a factor of $1/r^2$ can be assumed. In theory, the range can very accurately be computed. However, in practice, it is more difficult due to interferences and cost–accuracy tradeoffs. A more accurate approach is the TOF technique or modifications such as *time difference of arrival* (TDOA). Here, the time between emitting and receiving a signal is measured. Distances can be estimated by multiplying the measured times with the velocity of propagation. Best results at relatively low costs can be accomplished by using a transmission medium with a propagation speed far below the speed of light such as ultrasound. The second technique is *angulation*. For position calculation, angles are used instead of distances. Only two reference points are required for 3-D position detection. For the *lateration* technique, at least three points are needed.

In this paper, we concentrate on local positioning systems (LPSs), where the coordinates, in contrast to global positioning systems, are valid only in a local context. A number of different LPSs have been proposed during the last two decades. The system called *Active Badge* [8] is often claimed to be one of the first developments—it has been published in 1992. The AT&T research team placed single infrared (IR) receivers in different rooms and connected them to a central server. The idea was to locate persons who are equipped with active badges. Every 10 s, the badges emit an IR pulse with a globally unique identification number. Thus, it is possible to provide both absolute and symbolic location information about the people. However, the system does not know the exact position of a person, but in which room she/he currently is.

Three years prior to the Active Badge, a *physical-position-sensing* system had already been published [9]. The authors used a combination of ultrasound and IR sensors. The system to be localized, in this case a mobile robot at an unknown position, emits an active ultrasound chirp. Beacons placed in the environment can detect this signal, and after a predefined waiting time, the beacon replies to the chirp with an IR burst that contains its location. The distance between the active beacon and the robot is determined by the elapsed time interval. Using a certain number of distance measurements and the TOF lateration technique, a position can be calculated. The gradient or Newton–Gauss method can be applied to the erroneous data to achieve higher accuracy. In reported experiments, an accuracy of less than 10 cm has been achieved. Similar to the active badge system, IR localization is very sensitive to the current light conditions and scalability is an issue.

In 1991, Leonard and Durrant-Whyte [10] used corners, walls, and other distinctive objects as passive beacons. The shape and, therefore, the object itself is detected by the use of an ultrasound distance analyzer. A map of the geometric beacon locations has to be known by the robot *a priori*. The proximity technique allows the vehicle to roughly estimate its location. In addition, the robot uses odometry and an extended Kalman filter for enhancing the accuracy of the location estimation. This technique can only be applied if 2-D positioning is needed. Aside from other effects, in 3-D space, the number of required measurements for beacon detection would be very high.

Angulation techniques are frequently based on optical measurements such as using a digital charge-coupled device (CCD) camera and appropriate pattern recognition algorithms. Such processes are very slow and consume much power. Hence, Salomon *et al.* [11] used an analog position-sensitive device and equipped the object to be localized with an IR emitter. Using these tools, an angle can be calculated. The power consumption on the receiver side is less than 60 mW; however, the possible detection angle of the system is very small.

RADAR [12] uses the signal strength and signal-to-noise-ratio of wireless local area network (LAN) for indoor position sensing. Similarly, Bulusu *et al.* [13] provide a solution for outdoor environments. Both approaches use the scene analysis technique. The reference points either broadcast their locations,

or they are stored in a database. Depending on the beacons in range, the location is computed (fingerprint). Reflected signal waves make it very hard to provide an accurate position, particularly for indoor usage. Still, an accuracy of about 4 m can be achieved. Again, this technique works only well for 2-D localization.

Beep [14] is another approach that relies on sound-based TOF lateration. In contrast to other implementations, an audible sound is used instead of ultrasound. This allows users to rely on personal digital assistants (PDAs) or cell phones as a receiver. One minor disadvantage is that the used hardware was not built for accurate time measurements. This fact is also reflected in the position accuracy: errors larger than 1 m have been observed.

A very precise LPS is 3-D-LOCUS [15], which is an ultrasound-based system with a subcentimeter accuracy. Within their work, the authors compared the two commonly known emission approaches (centralized and private) with a third bidirectional strategy. The latter case not only boosts the accuracy but also minimizes environmental effects.

In practice, the clock synchronization of all involved controllers is often not possible. In such cases, TDOA techniques have to be used. The lack of knowledge of the time of departure of a signal can be compensated for by taking not only the position as a variable but also the time. Only one more reference point is needed to solve the resulting equations. Mahajan and Walworth [16] developed a closed-form solution for this kind of problem.

Lin *et al.* [17] utilized fuzzy logic for the position estimation. To meet their accuracy requirements and timings, they have to use an industrial personal computer with standard C++. The test bed consists of a well-defined number of ultrasound transducers at certain positions. For the autonomous case, these conditions are hard to achieve.

Some years after the active badge, the same institution (AT&T Laboratories Cambridge, formerly Olivetti Research Limited) proposed a new localization system called *Active Bat* [18]. It relies on ultrasound-based TOF lateration. A bat, which is carried around by a person, sends an ultrasound chirp to a grid of ceiling-mounted receivers. Simultaneously, the receivers are synchronized and reset by a radio packet that is also transmitted by the bat. All measured distances are forwarded to a central computer, where the position calculations take place. An accuracy of 9 cm has been achieved. The scalability is limited by the central computer and wires to all the ultrasound receivers. That weakness has been addressed with the *Cricket* system [4]. All the wires have been replaced by wireless communication and distributed location calculation (on the node to be localized). The localization is initiated with a localization request radio packet. Because this packet does not include any identifier and the location computation is performed on the object itself, location privacy is provided. However, as the position-sensing time intervals can get very big, the solution is not suitable for continuous real-time localization. Finally, Hazas and Ward [19] merged location privacy and a high localization rate by utilizing broadband ultrasound hardware and direct sequence spread spectrum/code-division multiple access (DS/CDMA).

We provide a continuous self-localization technique based on ultrasonic distance measurements, which can self calibrate in case of measurement errors. The key idea is to use a modified Kalman filter that not only predicts positions but also allows us to recover from severe measurement errors. All the necessary calculation can be performed even by sensor nodes with limited computational power.

III. Localization Procedure

This section outlines the procedure of computing position information out of gathered distance measurements. We rely on ultrasonic distance estimation for TOF-based lateration. Technical details of our hardware system are provided in Section V.

A. Preliminaries

We assume to start with a set of n tuples T_i, each consisting of a distance d_i to a reference point with a known position and the coordinates of this point $\overrightarrow{x_i}$. We have

$$T_i = (d_i, \overrightarrow{x_i}) : \overrightarrow{x_i} = (x_i, y_i, z_i)^T, \qquad i \in [1, n]. \quad (1)$$

The trilateration problem can be solved for the unknown position, $\overrightarrow{x} = (x, y, z)^T$, in different ways. Theoretically, the problem can be solved by a closed mathematical expression, as shown in Equation (2). However, in practice, it is impossible to solve those n equations at once due possibly error-prone measurements. We have

$$(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2 = d_i^2; \qquad i \in [1, n]. \quad (2)$$

Several iterative optimization algorithms exist for the problem. For example, Foy [20] uses Taylor series estimation. At least for 2-D problems, the method converges to a reasonably good solution within a few iterations. Another common approach is the use of an extended Kalman filter [21]. Apart from high computation costs, one of its biggest advantages is that this technique can be applied with at least one measurement. In general, all techniques perform an iterative computation and finally present a result with an (almost) negligible error. Closed and, therefore, simple and fast solutions are more suitable for low-power sensor nodes. Abroy *et al.* [22] present such a noniterative solution, but with tremendous restrictions in terms of scalability and variability. Exactly three reference points, precisely oriented to each other, are required: the coordinates have to be $\overrightarrow{x_1} = (0, 0, 0)^T$, $\overrightarrow{x_2} = (x_2, 0, 0)^T$, and $\overrightarrow{x_3} = (x_3, y_3, 0)^T$. To apply this system to a general case, a coordinate transformation (offset and rotation) would be needed. However, due to the nonnegligible computational effort, this method cannot be applied in many scenarios.

B. Position Calculation

One common feature of all indoor location systems attracted our attention. Given that all reference points are mounted to the ceiling, the wall, or the floor, they all have one coordinate in common. Let us denote this coordinate as the z-coordinate. We exploit this information for a closed position calculation.

First, a distribution of all tuples T_i into m subsets S_j to pairs of three different points must be done. The precise subset generation method will be explained in Section III-C. For the moment, we assume that we have m subsets that fulfill the condition that all z-coordinates within a subset S_j of all tuples T have to be equal. We have

$$S_j \subseteq T | \forall \overrightarrow{x_i} \in S_j : z_i = c_j, \qquad c_j \in \mathbb{R} \text{ and } \|S_j\| = 3. \quad (3)$$

Furthermore, it must be defined *a priori* whether the object to be localized is above the selected c_j, i.e., $z \geq c_j$, or below, i.e., $z \leq c_j$.

Then, we can compute m possible coordinates for the unknown object out of the m subsets. Using a set of three single equations based on (2) and taking the characteristics of each subset S_j into account, we can form a linear equation system, i.e.,

$$\begin{aligned} & A\overrightarrow{x} = \overrightarrow{b} : A \in \mathbb{R}^{2\times 2}, \qquad \overrightarrow{x} \in \mathbb{R}^2, \quad \overrightarrow{b} \in \mathbb{R}^2 \\ & A = 2 \cdot \begin{bmatrix} x_3 - x_1 & y_3 - y_1 \\ x_3 - x_2 & y_3 - y_2 \end{bmatrix} \\ & \overrightarrow{x} = (x, y)^T \\ & \overrightarrow{b} = \begin{pmatrix} \left(d_1^2 - d_3^2\right) + \left(x_3^2 - x_1^2\right) + \left(y_3^2 - y_1^2\right) + \left(z_3^2 - z_1^2\right) \\ \left(d_2^2 - d_3^2\right) + \left(x_3^2 - x_2^2\right) + \left(y_3^2 - y_2^2\right) + \left(z_3^2 - z_2^2\right) \end{pmatrix}. \end{aligned} \quad (4)$$

This 2-D problem can easily be solved by applying Gaussian elimination.

Before the computation can be performed, some basic checking needs to be performed to test whether the system can be solved: Basically, the matrix A must have no rank defect (linearly independent vectors), i.e., $RgA \stackrel{!}{=} 2$. For ease of computation, it is sufficient to test whether the determinant of the matrix is not equal to zero, i.e., $\det A \neq 0$. Geometrically, this test can be interpreted as three reference points that span a plane.

For the computation of the x- and y-coordinates, only fast arithmetic operations are needed, such as addition, subtraction, and multiplication, which are available on low-cost microcontrollers. The z-coordinate can be generated in two ways. The easiest way is simply to measure it, which is straightforward using an ultrasound system. Alternatively, the already-computed values can be inserted in (2), which, however, requires a square root to be calculated on the used microcontroller.

Equation (3) restricts the z-coordinate of each subset to be equal. If this condition cannot be fulfilled, the algorithm is not applicable. This situation can be avoided using a coordinate transformation (rotation). After computing the position, a backtransformation into the original coordinate system is required, i.e.,

$$\overrightarrow{z} = \Theta(\overrightarrow{x}); \text{position algorithm}; \overrightarrow{x} = \Theta^{-1}(\overrightarrow{z}). \quad (5)$$

C. Subset Generation

In theory, one subset S_j, which contains three tuples T_i, would be sufficient for position estimation. However, taking measurement errors into account, more subsets are required. Let n be the number of collected tuples T_i (of reference points $\overrightarrow{x_i}$ and distances d_i). Then, $m = \binom{n}{3} = (n!/3! \cdot (n-3)!)$ disjunct subsets of three pairs can be computed. The number of possible subsets significantly increases with the number of reference points. In terms of scalability, it is not feasible to compute and evaluate all m subsets.

Casas *et al.* [23] investigated all kinds of ultrasonic measurement errors. They came up with an average rate of measurement failure of $P_{mf} = 30\%$. A position estimation can only be successful if at least one correct subset S_j is used for evaluation, where a correct subset corresponds to one that contains only accurate measurements. P_m denotes the probability that none of the chosen subsets is correct. The required number of subsets can be calculated as follows [23]:

$$m = \frac{\log(P_m)}{\log\left(1 - (1 - P_{mf})^3\right)}. \quad (6)$$

For example, 11 subsets are required if we accept a failure probability of $P_m = 1\%$. Furthermore, the authors suggest that Monte Carlo techniques should be applied to randomly pick m subsets. However, more information about the subsets could help improve the selection.

In general, subsets with geometric shapes that minimize the error rate of the position calculation should be preferred (e.g., regular or well-formed triangles). Thus, the basic idea is to generate and, subsequently, to qualify a subset. Then, it can be placed in a sorted list. This process continues until a certain threshold is reached for the mth element of the list. Finally, the first m elements in this list are then used for the position calculations.

We decided to use a weighted combination of the average measured distances, and the covered ground of the three points would be suitable. Both values are important for a well-formed but (mostly) nonregular tetrahedron (three reference points plus the unknown point). The base area of the figure is a triangle. Unfortunately, this case cannot be efficiently computed because of the necessary square root or trigonometric functions. Therefore, we used the cross product, $\hat{\overrightarrow{n}} = \overrightarrow{a} \times \overrightarrow{b}$ (with $\overrightarrow{a} = \overrightarrow{x_2} - \overrightarrow{x_1}$ and $\overrightarrow{b} = \overrightarrow{x_3} - \overrightarrow{x_1}$). Its length directly corresponds to the covered area; in particular, it represents not the area of the spanned triangle but of the corresponding parallelogram. Thus, a division by 2 results in the correct area.

According to (3), the base area is parallel to the xy plane; therefore, the cross product only contains a z-component, i.e.,

$$\overrightarrow{a} \times \overrightarrow{b} = \begin{pmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{pmatrix} \stackrel{(3)}{=} \begin{pmatrix} 0 \\ 0 \\ a_1 b_2 - a_2 b_1 \end{pmatrix}. \quad (7)$$

This length can be computed very fast, because only summations, subtractions, and multiplications are needed.

D. Position Estimation

Finally, the m possible positions [stored in $X(k+1) : X \in \mathbb{R}^{3\times m}$] have to be merged to one position $\overrightarrow{x}(k+1)$. The trivial

approach would be the calculation of the mean of all positions. However, outliers would significantly influence the result. Casas *et al.* [23] used an approach where a squared residual vector between all measured and all theoretical distances for each subset are computed. By taking the minimum median of the individual elements, the influence of the outliers vanishes. Unfortunately, the computational effort for this method increases with the number of reference points and, therefore, is not very scalable.

Thus, in the first step, we experimented with a refinement of the trivial concept. We implemented a recursive algorithm, which generates the mean of all positions and removed the position that is most distant to the mean. This step is repeated until a unique position is found. Unfortunately, we did not get satisfactory results due to the high rate of measurement errors of about 25%.

In the second step, we incorporated prior knowledge into the position estimator. The Casas method [23] provides localization without any state information. However, already-collected information could be exploited to gain better localization results. Thus, we split the estimation process into two steps similar to an extended Kalman filter. In the first step, we predict the current position $\overrightarrow{x_p}(k+1)$ using a state vector as follows:

$$\overrightarrow{x_p}(k+1) = \overrightarrow{x}(k) + \Delta\overrightarrow{x}(k, k-1) \cdot r \cdot \kappa(r) \tag{8}$$

$$r = \frac{\Delta t(k+1, k)}{\Delta t(k, k-1)}. \tag{9}$$

For this vector, in each step, we store the position and the localization time. The second step is slightly different from the original design of the filter. Usually, a motion model is required to improve the predicted position. A generally valid model for a hovering robot, which is on top of a highly unpredictable air bubble, as well as for an arbitrary moving target, is hard to find. However, the benefits of the possibly more accurate localization would not balance out the increased computational effort. This case particularly holds for our sensor network scenario.

Therefore, the new position $\overrightarrow{x}(k+1)$ is generated by selecting the nearest computed position to the predicted position out of the set $X(k+1)$ as

$$\overrightarrow{x}(k+1) = f\left(\overrightarrow{x_p}(k+1), X(k+1), \Delta t(k+1, k)\right). \tag{10}$$

If none of the computed positions is within a certain radius from the predicted position, all computed positions are considered erroneous and will be rejected. The radius grows with the elapsed time, and therefore, the probability of acceptance increases even if only false positions are obtained.

The more the time has elapsed since the last computation in relation to the last interval, the less reliable the prediction gets. The correction function $\kappa()$ in (8) has been designed to compensate for this effect. Usually, if an algorithm detects such a situation, then the state vector is reset, and an initial filter stage is reentered. Appropriate time intervals have to be configured for this purpose. By using the ratio between the localization attempts, this mechanism can be automated. Furthermore, the absolute computation frequency is not relevant. $\kappa()$ is a function of r [Equation (9)], which denotes the ratio of two time intervals: 1) the current time and 2) the time at which the last position has been accepted. $\kappa()$ is a simple function that returns 1 for values between 0 and 1. For greater values, the output slowly decreases to 0. Fig. 2 illustrates the prediction vector and the growing space of the position acceptance.

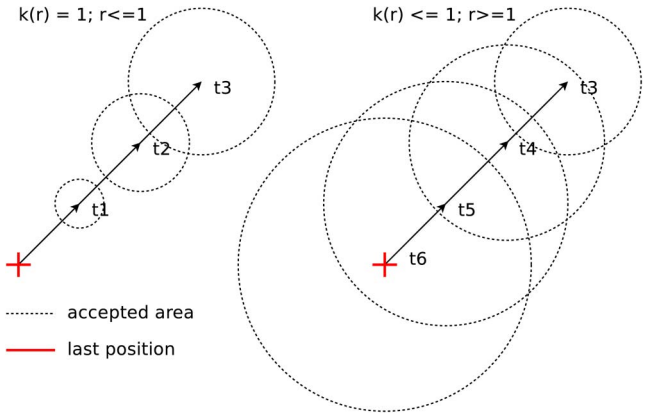

Fig. 2. Position prediction.

As shown on the left side, the prediction vector grows uninfluenced over time if the ratio r is smaller than 1, and therefore, $\kappa()$ is 1. Thus, $\kappa()$ does not influence the prediction. The right side shows the situation if the ratio r increases beyond 1. This case means that the last localization interval (i.e., the time between two accepted positions) was shorter than the elapsed time since the last position has been accepted. Now, $\kappa()$ is decreased, because at this time, a proper prediction based on the movement during the last interval cannot be guaranteed.

We achieved very good and fast results using this estimation technique. However, it can happen that a wrong position is accepted, e.g., if all taken measurements are wrong and, therefore, all possible positions are also incorrect. In this case, the position estimator takes the nearest of these wrong positions if it is within the accepted area. Therefore, an invalid state and the associated positions are stored. Fortunately, the localization technique is self correcting. As soon as the object moves (erroneous measurements are fluctuating) and at least one correct possible position is continuously calculated, the state will again become accurate within a few cycles.

E. Complete Algorithm

The complete procedure for obtaining a position is summarized in Algorithm 1. As long as the position information is needed, the algorithm stays in the outer loop. After the measurement for the current time slot is completed and the results are locally transferred to the sensor node, the master node starts to collect the data from the slave nodes by sending out an agent frame. After a certain time has passed and if enough tuples T could be collected from the participating sensor nodes, the actual computation takes place. At first, m subsets S are generated (Section III-C). Out of those m possible positions, X are calculated (Section III-B). The position $\overrightarrow{x}$

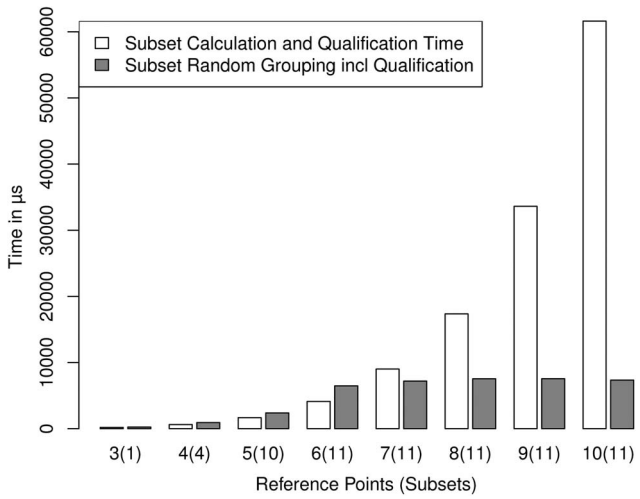

Fig. 3. Subset calculation.

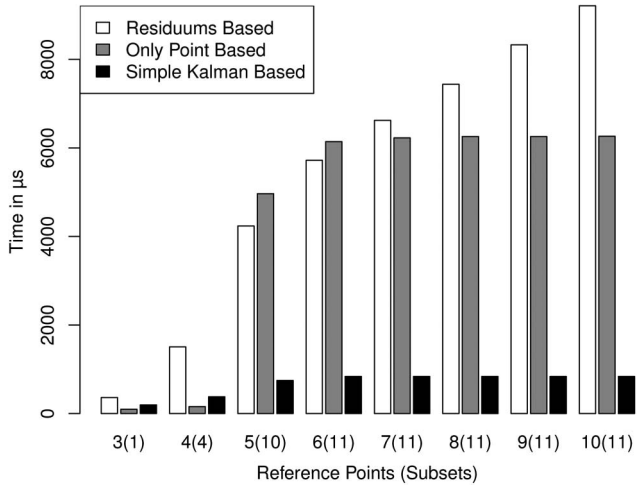

Fig. 4. Position estimation.

is finally estimated by applying the position estimator to all positions in X (Section III-D).

Algorithm 1: Localization algorithm
while running **do**
do measurements
collect i tuples T
if $i \geq 3$ **then**
generate m subsets S from T
generate m possible positions X from S
estimate the current position from X
end if
end while

IV. Localization Performance

Scalability is one of the biggest issues in the context of sensor networks. As a proof of concept for our localization algorithm, we implemented the system and evaluated it in a laboratory scenario. In particular, we used the Sun Small Programmable Object Technology (SPOT) sensor node platform [24] that runs JavaME as the host operating system. We first estimated the computational performance of the localization algorithm. In the next section, we will discuss the applicability for real-time localization of our flying quadrocopter.

One of the key issues is the creation of the subsets. Fig. 3 shows the required time of the grouping for different numbers of reference points and subsets. For reasons explained in Section III-C, we limited the number of subsets to 11. Independent of the number of reference points, an upper boundary for the classification can be given. The limitation of subsets also implicitly restricts the position vector calculation time to an upper boundary. Thus, not every possible position needs to be calculated: Only positions from subsets that meet a certain threshold in the qualification are considered. Furthermore, those calculations benefit from the grouping algorithm, because no further test whether (4) can be solved has to be performed.

In Fig. 4, all the tested timings of the position estimation algorithms are depicted. As aforementioned, the residuum-based method [23] scales approximately linear to the number of reference points. Similarly, the only-point-based recursive algorithm does not scale well, the computation costs are very high, and the demands for accuracy cannot be met. The Kalman-based predictive method gives accurate and quick results.

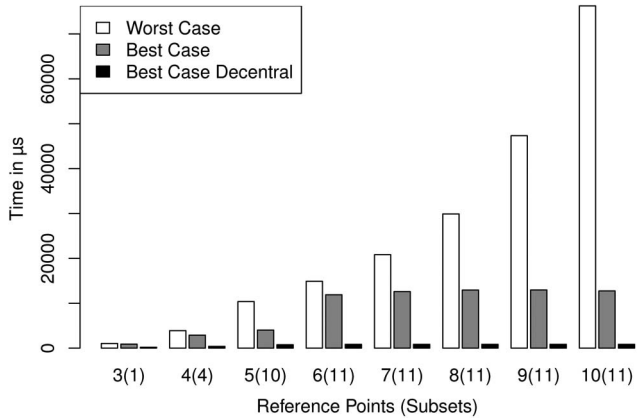

Fig. 5. Total localization time.

Finally, Fig. 5 shows the total computation time. The worst case scenario is a combination of the techniques that are not bounded in computational time. All subsets are computed, and the residual-based position estimation was applied to the best 11 subsets. In the best case scenario, only techniques with a bounded computational time are used. Therefore, an upper bound for the localization algorithm can be given, independent of the number of used reference points. This condition is important to fulfill the real-time specification. The best case decentral scenario describes the absolute minimal computational time consumption for the initiator of the localization if subset grouping and position vector calculations are distributed on the entire sensor network. Unfortunately, the overhead of the communication latency is far too big to benefit from it, at least using our available hardware.

V. Test System

In this section, we describe our localization system for the real-time control of a quadrocopter. We also discuss the practical implementations of the developed algorithms.

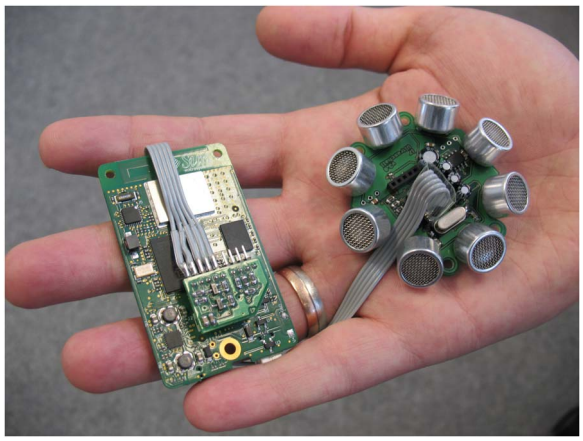

Fig. 6. Ultrasonic measurement system with the Sun SPOT node.

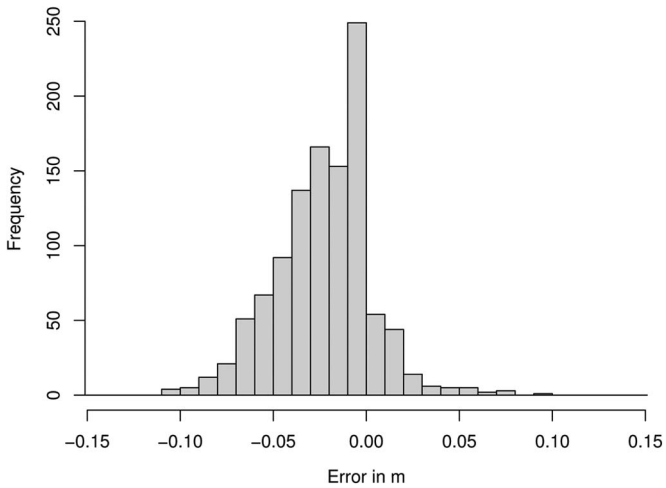

Fig. 7. Localization accuracy. The quadrocopter hovers over the ground, and the height error is plotted.

Fig. 6 shows the latest version of our ultrasonic measurement system, including the sensor node. Despite the classical master–slave topology, we decided for a hybrid measurement architecture. Whether a device is a master (transmitter) or a slave (receiver) is completely hardware independent and can be controlled on the application level. The measurement accuracy of the system is ± 2 cm. In [6], we provide more detailed information about the sensors. The detection field of the system is designed to be a slightly flattened hemisphere with a radius of more than 5 m. Thus, the reference points on the floor can detect not only the flying object but also each other (this architecture is depicted in Fig. 1). This way, it is possible to automatically span up the grid by attaching the reference points on top of mobile robots. Another advantage of a flying active beacon, as aforementioned, is that, by sensing the TOF of its own active chirp, the altitude of the object can be computed without the help of the localization infrastructure.

To measure the localization accuracy, we arranged one reference point in each corner of a square; therefore, a total of four reference points are used. The length of the edges was 2 m. The object randomly hovers in a square of about 3 m of edge length and at an altitude of 0.5–2.5 m over the reference square. Fig. 7 shows the measurement results. The histogram shows the difference between the measured (using the ultrasonic device) and the computed altitude (using the localization system). An accuracy of ± 10 cm can be achieved, with a confidence of 98%. Faulty measurements that result in highly fluctuating positions are completely discarded by the position estimation process.

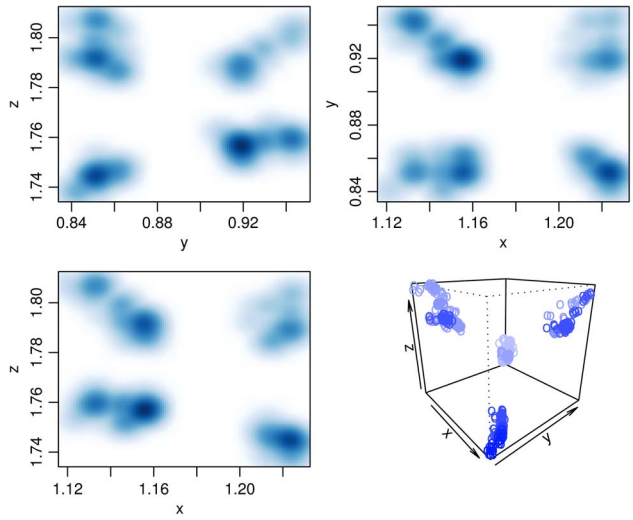

Fig. 8. Localization accuracy. The quadrocopter is fixed in a position, and the measured x-, y-, and z-coordinates are plotted.

For the measurements shown in Fig. 8, we placed the four-rotor robot at an arbitrary but fixed position over the detection field. A moving person within this square sometimes covered the direct line of sight between the measurement points. It is shown that there are four centers of gravity. Each subspace is the region for the computed position of one of the four possible subsets. Within this space, the maximum variance is about ± 2 cm. The estimated position is normally confined to one of those regions. However, as soon as the used subset is missing, the estimated point jumps to another subspace. The temporary vanishing of a subset can have two main reasons. First, one of the measurements was wrong, and therefore, the position was very far away. Second, the wireless communication may be disrupted.

The generation of the regions is based on systematic errors of the reference points' positions. In our tests, we ensured an accuracy of about ± 3 cm. With increasing deployment accuracy of the reference points, the resulting regions merge into a single one.

VI. Communication

In this section, we discuss relevant communication issues in the network. Among the most important criteria, we looked into availability and real-time capabilities.

For communication, we used the on-board radio capabilities of Sun SPOT. It contains an IEEE 802.15.4 [25] compatible interface (Chipcon CC2420) for wireless communication. Due to its low-power specifications, the emitted signal is not very strong. Of course, this condition influences the transmission range. The disadvantage is even amplified by the lack of an external antenna. For our tests, we used channel 26. Due to Federal Communications Commission (FCC) regulations, a maximum output power of -3 dBm (0.5 mW) is allowed on that channel. By changing the channel, the power could be increased

to 0 dBm (1.0 mW). However, this case would only shift the problem to bigger distances. Therefore, we left the transmitting power at -3 dBm to show the problems using a smaller environment.

After performing a measurement, the measured values are distributed within the sensor network. These data have to be transmitted back to the active beacon for location computation. Our first attempt was to dispatch a request and to have all sensor nodes with relevant data reply to this using the Aloha technique [26]. However, in our aforementioned environment, the probability of getting three or more measurements responses was less than 65%. Experiments showed that the probability of receiving a broadcast packet is less than 80% if the receiver is at least 2 m away. Relying on unicast packets is not feasible, because if a packet cannot be delivered, the whole transmission unit can be blocked for more than 10 s. The impact on the flying object would be dramatic. Therefore, we developed a custom agent user-level protocol based on broadcasts. We intended to create a protocol that is not restricted to IEEE-802.15.4-compatible physical interfaces but can be applied to other general-purpose radio systems.

The initial frame is broadcast by the active beacon without any payload. One field of the frame contains an identifier. If the agent travels very long, the initiated node will no longer accept the packet. This condition is necessary to meet the real-time constraints. Three things can happen if a node receives an agent frame. If the agent has already passed through the system, the agent is dropped. If the agent is new to the system, then a departure time is generated, depending on its options if the time to live (TTL; hop counter) is not exceeded. In addition, relevant measurements are added before the departure. If an agent arrives with the same identifier as one in the departure queue (duplicate agent), the new information from the recently arrived agent is copied to the one in the queue. The rest (duplicate information) is dropped. The initiator acts similar to the rest of the network, except that it does not retransmit the agent but holds all gained data in a buffer until the lifetime of an agent is reached. After that time, the data can be released for further processing. Hence, a deterministic time for the availability of the data can be given. The data collection is possible on not only the initiator side but also on all other systems. All measured information can be made accessible on every sensor node in range without extra radio load.

Fig. 9 shows a histogram plot of the initial communication experiment. As aforementioned, only four reference points were available for testing. Summing up the probability for three and four reference points per cycle gives the probability of getting enough measurements for the localization computation. Therefore, in at most 67% of all cases, a position can be computed. Furthermore, in only 30% of all cases, more than one position for detecting and compensating measurement errors can be computed.

In Fig. 10, the results for the agent protocol are depicted. Here, the sum of probabilities for three and four reference points per cycle is higher than 95%. Furthermore, the probability for four reference points has significantly increased to about 82%. This result allows for much more precise position estimation.

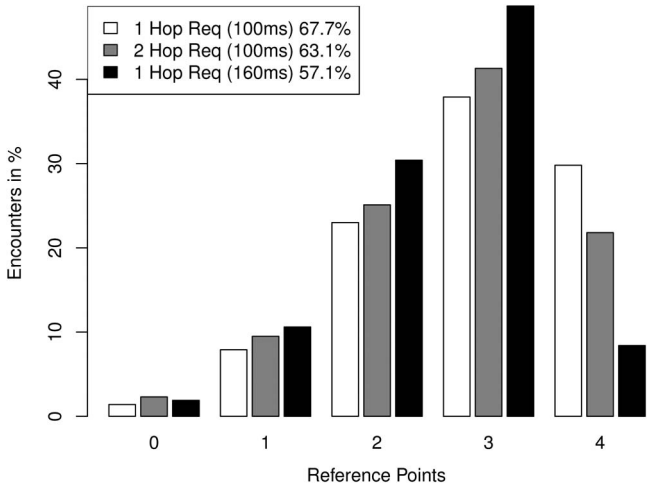

Fig. 9. Availability in the case of using the request-reply technique with broadcast.

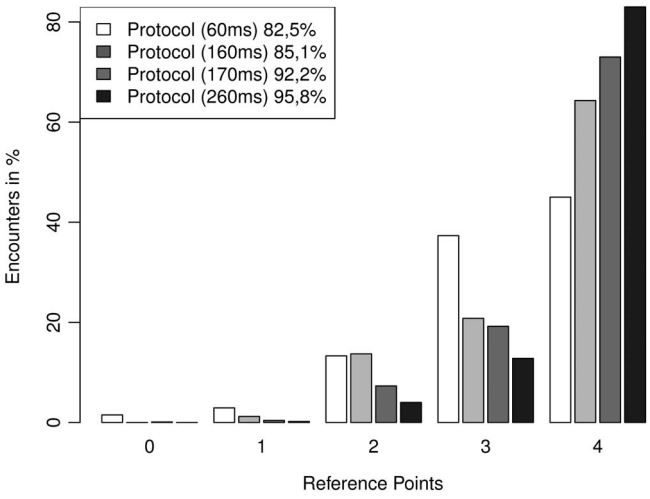

Fig. 10. Availability in the case of using the agent protocol.

VII. Conclusion

We have investigated the problem of continuous indoor localization for flying autonomous robots. In contrast to ground-based robots, any waiting until position measurements have been completed, or taking advantage of additional support systems such as odometry is not possible in this case. Thus, real-time localization is needed, which must also take weight constraints into account.

Considering these requirements, we have developed an algorithmic procedure that can perform real-time localization based on possibly error-prone distance measurements even on a sensor node with limited computational power. The basic assumption is that one coordinate of the reference points needs to be equal. Without loss of generality, we set the z-coordinates to a constant value. This approach allows a closed mathematical calculation that is even possible to be performed by low-resource sensor nodes. If, however, a coordinate transformation needs to be executed, the localization algorithm suffers from the computational complexity of this transformation. We implemented and evaluated the algorithm in our laboratory. The results demonstrate the feasibility of the solution. We consider our ultrasound lateration technique a necessary step

for a completely autonomous operation of flying robots in indoor environments. To improve the measurement accuracy, further research is needed to compensate for the drift of the ultrasound path caused by the constant air flow of the rotors, as illustrated in a study by Prieto *et al.*

REFERENCES

[1] P. Tomei, "Adaptive PD controller for robot manipulators," *IEEE Trans. Robot. Autom.*, vol. 7, no. 4, pp. 565–570, Aug. 1991.

[2] W. Liao, J. Sheu, and Y. Tseng, "GRID: A fully location-aware routing protocol for mobile ad hoc networks," *Telecommun. Syst.*, vol. 18, no. 1–3, pp. 37–60, Sep. 2001.

[3] A. Smith, H. Balakrishnan, M. Goraczko, and N. Priyantha, "Tracking moving devices with the Cricket location system," in *Proc. 2nd ACM MobiSys*, Boston, MA, 2004, pp. 190–202.

[4] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The Cricket location-support system," in *Proc. 6th ACM MobiCom*, Boston, MA, Aug. 2000, pp. 32–43.

[5] J. Eckert, F. Dressler, and R. German, "Real-time indoor localization support for four-rotor flying robots using sensor nodes," in *Proc. IEEE Int. Workshop ROSE*, Lecco, Italy, Nov. 2009, pp. 23–28.

[6] J. Eckert, K. Koeker, P. Caliebe, F. Dressler, and R. German, "Self-localization-capable mobile sensor nodes," in *Proc. IEEE Int. Conf. TePRA*, Woburn, MA, Nov. 2009, pp. 224–229.

[7] J. Hightower and G. Borriello, "A survey and taxonomy of location systems for ubiquitous computing," Univ. Washington, Washington, DC, Tech. Rep. UW-CSE 01-08-03, Aug. 2001.

[8] R. Want, A. Hopper, and V. Gibbons, "The active badge location system," *ACM Trans. Inf. Syst.*, vol. 10, no. 1, pp. 91–102, Jan. 1992.

[9] C. Durieu, H. Clergeot, and F. Monteil, "Localization of a mobile robot with beacons taking erroneous data into account," in *Proc. IEEE Int. Conf. Robot. Autom.*, Scottsdale, AZ, May 1989, pp. 1062–1068.

[10] J. Leonard and H. Durrant-Whyte, "Mobile robot localization by tracking geometric beacons," *IEEE Trans. Robot. Autom.*, vol. 7, no. 3, pp. 376–382, Jun. 1991.

[11] R. Salomon, M. Schneider, and D. Wehden, "Low-cost optical indoor localization system for mobile objects without image processing," in *Proc. IEEE Conf. ETFA*, Praha, Czech Republic, Sep. 2006, pp. 629–632.

[12] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *Proc. 19th IEEE INFOCOM*, Tel-Aviv, Israel, Mar. 2000, pp. 775–784.

[13] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low-cost outdoor localization for very small devices," *IEEE Pers. Commun.*, vol. 7, no. 5, pp. 28–34, Oct. 2000.

[14] A. Mandal, C. Lopes, T. Givargis, A. Haghighat, R. Jurdak, and P. Baldi, "Beep: 3D indoor positioning using audible sound," in *Proc. 2nd IEEE CCNC*, Las Vegas, NV, Jan. 2005, pp. 348–353.

[15] J. C. Prieto, A. R. Jiménez, J. Guevara, J. L. Ealo, F. Seco, J. O. Roa, and F. Ramos, "Performance evaluation of 3D-LOCUS advanced acoustic LPS," *IEEE Trans. Instrum. Meas.*, vol. 58, no. 8, pp. 2385–2395, Aug. 2009.

[16] A. Mahajan and M. Walworth, "3D position sensing using the differences in the time-of-flight from a wave source to various receivers," *IEEE Trans. Robot. Autom.*, vol. 17, no. 1, pp. 91–94, Feb. 2001.

[17] H.-H. Lin, C.-C. Tsai, and J.-C. Hsu, "Ultrasonic localization and pose tracking of an autonomous mobile robot via fuzzy adaptive extended information filtering," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 9, pp. 2024–2034, Sep. 2008.

[18] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster, "The anatomy of a context-aware application," in *Proc. 5th ACM MobiCom*, Seattle, WA, Aug. 1999, pp. 59–68.

[19] M. Hazas and A. Ward, "A high-performance privacy-oriented location system," in *Proc. IEEE Int. Conf. PerCom*, Dallas-Fort Worth, TX, Mar. 2003, pp. 216–223.

[20] W. Foy, "Position location solutions by Taylor series estimation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-12, no. 2, pp. 187–194, Mar. 1976.

[21] L. Kleeman, "Optimal estimation of position and heading for mobile robots using ultrasonic beacons and dead reckoning," in *Proc. IEEE Int. Conf. Robot. Autom.*, Nice, France, May 1992, vol. 3, pp. 2582–2587.

[22] M. Abreu, R. Ceres, L. Calderon, M. Jimenez, and P. Gonzalez-de Santos, "Measuring the 3D position of a walking vehicle using ultrasonic and electromagnetic waves," *Sens. Actuators A: Phys.*, vol. 75, no. 2, pp. 131–138, May 1999.

[23] R. Casas, A. Marco, J. Guerrero, and J. Falco, "Robust estimator for non-line-of-sight error mitigation in indoor localization," *EURASIP J. Appl. Signal Process.*, vol. 2006, pp. 1–8, 2006.

[24] R. Smith, "SPOTWorld and the Sun SPOT," in *Proc. 6th Int. Conf. Inf. Process. Sens. Netw.*, Cambridge, MA, Apr. 2007, pp. 565–566.

[25] Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs), IEEE Standard 802.15.4-2006, 2006.

[26] N. Abramson, "The Aloha System: Another alternative for computer communications," in *Proc. AFIPS Joint Comput. Conf.*, Houston, TX, Nov. 1970, pp. 281–285.

Juergen Eckert was born in Weiden i.d.OPf., Germany, in 1983. He received the B.Sc. and M.Sc. degrees (with distinction) in computational engineering in 2006 and 2008, respectively, from the University of Erlangen–Nuremberg, Erlangen, Germany, where he is currently working toward the Ph.D. degree, under the supervision of Professor Reinhard German, with the Computer Networks And Communication Systems Group in the Department of Computer Science.

His main research interests include robotics and wireless sensor networks.

Reinhard German received the Diploma degree in computer science and the Ph.D. and Habilitation degrees from the Technical University of Berlin, Berlin, Germany, in 1991, 1994, and 2000, respectively.

Then, he joined the Department of Computer Science, University Erlangen–Nuremberg, Erlangen, Germany first as an Associate Professor (system simulation) and, since 2004, as a Full Professor (computer networks and communication systems), where he is currently the Head of department. His research interests include model- and measurement-based performance analysis, modeling and simulation paradigms and tools, numerical analysis of Markovian and non-Markovian models, vehicular communications, and autonomous sensor/actuator networks.

Falko Dressler (S'02–M'04–SM'08) received the M.Sc. and Ph.D. degrees from the University of Erlangen–Nuremberg, Erlangen, Germany, in 1998 and 2003, respectively.

In 2003, he joined the Computer Networks and Internet Group, University of Tuebingen, Tuebingen, Germany. Since 2004, he has been with the Computer Networks and Communication Systems Group, University of Erlangen–Nuremberg. He is an Assistant Professor with the Department of Computer Sciences, University of Erlangen–Nuremberg. He teaches self-organizing sensor and actor networks, network security, and communication systems. He was a Guest Editor of special issues on self organization, autonomic networking, and bio-inspired computing and communication for the *Elsevier Ad Hoc Networks* and *Springer Transactions on Computational Systems Biology* (TCSB). He is an Editor for journals such as the *Elsevier Ad Hoc Networks* and *ACM/Springer Wireless Networks* (WINET). He published two books, including *Self-Organization in Sensor and Actor Networks* (Wiley, 2007). His research activities are focused on self-organizing networks that address issues in wireless ad hoc and sensor networks, intervehicular communication systems, bio-inspired networking, and adaptive network security techniques.

Dr. Dressler is a Senior Member of the Association for Computing Machinery Special Interest Group on Mobility of Systems, Users, Data, and Computing (ACM SIGMOBILE) and a Member of GI/ITG Kommunikation und Verteilte Systeme (KuVS; Real Time). He was a Guest Editor of special issues on self organization, autonomic networking, and bio-inspired computing and communication for the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS (JSAC). Aside from chairing a number of workshops associated to high-level conferences, he regularly acts in the Technical Program Committee (TPC) of leading networking conferences such as IEEE International Conference on Computer Communications (INFOCOM), IEEE International Communications Conference (ICC), IEEE Global Communications Conference (GLOBECOM), and IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS).