Real-time Indoor Localization Support for Four-rotor Flying Robots using Sensor Nodes

Juergen Eckert, Falko Dressler and Reinhard German
Computer Networks and Communication Systems
Department of Computer Science 7, University of Erlangen, Germany
{juergen.eckert, dressler, german}@informatik.uni-erlangen.de

***Abstract*—Flying four-rotor robots (quadrocopters) are onboard sensor controlled systems. In comparison to classical monorotor objects (helicopters), quadrocopters can be piloted with a much lower effort. However, lateral drifts can not be compensated only referring to the built-in sensors. The detection of such drifts is strongly necessary for indoor operation – without corrections a quadrocopter would quickly cause a collision. In order to compensate the dislocation, an additional indoor positioning system is needed. In our work, we provide a framework for time-of-flight based localization systems relying on ultrasonic sensors. It is optimized for use in sensor nodes with low computational power and limited memory. Nevertheless, it offers scalability and high accuracy even in case of single erroneous measurements. We implemented the system in our lab using ultrasound sensors that are light enough to be carried around by the flying object. Using this real-time localization system, a position controller can be implemented to maintain a given position or course.**

***Index Terms*—Indoor localization, flying robot, sensor network, ultrasound**

I. Introduction

Flying four-rotor robots are similar to helicopters. In contrast to mono-rotor systems, these so-called quadrocopters usually provide more sensors and more robust controllers. 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. In such a way devices can easily be piloted by other digital systems such as a sensor network. By only controlling the pitch and the roll angles, the current position cannot be obtained. The quadrocopter always hovers on top of an air cushion. Thus, any minimal measurement error or any airflow may cause a drift to a random direction. The system remains highly in-stable w.r.t. position maintenance. Angle corrections must be permanently applied and more than on board instruments need to be used to keep the flying robot in position.

Figure 1 shows the scenario. A quadrocopter is relying on an external positioning system to continuously update its system parameters. In general, there are many cases in which applications benefit from getting more accurate positioning information. A discussion of preferences for systems using active or passive mobile devices can be found in [1]. If privacy is an issue, passive localization systems should be preferred. For example, the infrastructure of the Cricket system [2] has no knowledge about the current position of any mobile device.

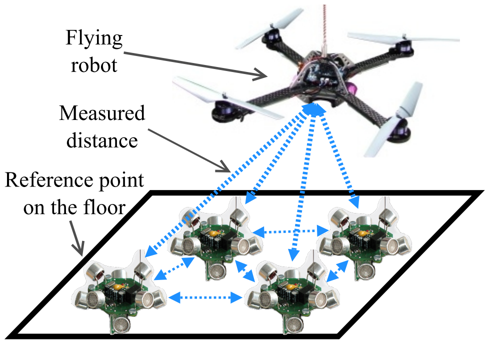

Fig. 1. Four-rotor flying robot hovers over reference points

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 is a strong limitation because a set of measurements can take up to several hundred milliseconds. In our scenario, the object to be localized is flying. That 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 receives it simultaneously. 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 problems. Unfortunately, no system (neither commercial nor academic) fulfills all the requirements for localizing flying quadrocopters. Real-time localization is frequently an issue, for example some systems rely on an iterative position estimation. Furthermore, many systems are simply too 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. We implemented a system based on ultrasonic distance measurements that is lightweight and can be carried by our quadrocopter. In summary, we not only provide a framework for our chosen scenario but also for other cases of real-time indoor localization. More detailed information can be found in our technical report [3].

The rest of the paper is organized as follows. Section II surveys the state of the art of localization systems. In Section III, we present the mathematical background of our localization system. Then, Section IV presents some insights into the performance of the system. The test system is finally described in Section V. Finally, Section VI concludes the paper.

II. Related Work

A number of different *Local Positioning Systems* have been proposed during the last two decades. A system called the *Active Badge* [4] is often claimed to be one of the first developments – it has been published in 1992. The AT&T research team placed single infra-red (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 $10\,\mathrm{s}$, those 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.

Already three years earlier, a *physical position sensing* system has been published [5]. The authors used a combination of ultrasound (US) and IR sensors. The system to be localized, in this case a mobile robot at an unknown position, emits an active US chirp. Beacons placed in the environment can detect this signal and, after a pre-defined waiting time, the beacon replies to the chirp with an IR burst containing 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 time-of-flight (TOF) lateration technique, a position can be calculated. The gradient or Newton-Gauss method can be applied to the erroneous data in order to achieve higher accuracy. In reported experiments, an accuracy of less than $10\,\mathrm{cm}$ has been achieved. Similarly to the active badge system, the IR localization is very sensitive to the current light conditions. Also, both systems do not scale very well.

In 1991, Leonard and Durrant-Whyte [6] used corners, walls, and other distinctive objects as passive beacons. The shape, and therefore the object itself, is detected by the use of an US distance analyzer. A map of the geometric beacon locations had 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-dimensional positioning is desired. Besides other effects, in 3-dimensional space the number of required measurements for beacon detection would be too huge.

Angulation techniques are frequently based on optical measurements such as using a digital CCD camera and appropriate pattern recognition algorithms. Such processes are extremely time and power consumptive. Hence, Salomon et al. [7] used an analogue position-sensitive device and equipped the object to be localized with an infrared emitter. Using these tools, an angle can be calculated. The power consumption on the receiver side is less than $60\,\mathrm{mW}$, however, the possible detection angle of the system is very small.

RADAR [8] uses the signal strength and signal-to-noise-ratio of wireless LAN for indoor position sensing. Similarly, Bulusu et al. [9] provide a solution for outdoor usage. Both approaches use the scene analysis technique. The reference points are either broadcasting 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, especially for indoor usage. Yet still an accuracy of about $4\,\mathrm{m}$ can be achieved. Again, this technique works only well for 2-dimensional localization.

Beep [10] is another approach relying on sound-based TOF lateration. In contrast to other implementations, audible sound is used instead of ultrasound. This allows the usage of PDAs or cell phones as a receiver. A slight disadvantage, besides the hearable measurement, is that the used hardware was not built for accurate time measurements. This fact is also reflected in the position accuracy: errors larger than $1\,\mathrm{m}$ have been observed.

In practice, clock synchronization of all involved controllers is often not possible. In such cases, time-difference-of-arrival techniques have to be used. The lack of knowing the time of departure of a signal can be compensated 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 [11] give a closed form solution for this kind of problem.

About seven years after the active badge, the same group proposed a new localization system called *Active Bat* [12]. It relies on US based TOF lateration. A bat, which is carried around by a person, sends an US 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 $9\,\mathrm{cm}$ has been achieved. The scalability is limited by the central computer and wires to all the US receivers. That weakness has been addressed with the *Cricket* system [2]. 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. As this packet does not include any identifier and because the location computation is performed on the object itself, location privacy is provided. However, as the position sensing time intervals can get too big, the solution is not suitable for continuous real-time localization.

III. Mathematical Procedure

This section covers the procedure of computing position information out of gathered distance measurements. We rely on US distance estimation for TOF based lateration. The technical details are depicted in Section V.

A. Preliminarities

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 $\overrightarrow{x_i}$:

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

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 to error-prone measurements.

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

Several iterative optimization algorithms exist for the problem. For example, Foy [13] uses a Taylor-series estimation. At least for 2-dimensional problems, the method converges to a good solution within a few iterations. Another common approach is the use of an extended Kalman filter [14]. Abroy and co-workers [15] present a non-iterative solution, however, 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 $\overrightarrow{x_3} = (x_3, y_3, 0)^T$. In order to apply this system to a general case, a coordinate transformation (offset and rotation) would be needed. Because this requires non-negligible 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 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 later in Section III-C. For the moment, we assume 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:

$$S_j \subseteq T \mid \forall \overrightarrow{x_i} \in S_j : z_i = c_j, c_j \in \mathbb{R} \text{ and } \|S_j\| = 3 \tag{3}$$

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 from (2) and taking the characteristics of each subset S_j into account, we can form a linear equation system:

$$A\overrightarrow{x} = \overrightarrow{b} : A \in \mathbb{R}^{2\times 2}, \overrightarrow{x} \in \mathbb{R}^2, \overrightarrow{b} \in \mathbb{R}^2 \tag{4}$$

$$A = 2 \cdot \begin{bmatrix} x_3 - x_1 & y_3 - y_1 \\ x_3 - x_2 & y_3 - y_2 \end{bmatrix}$$
$$\overrightarrow{x} = (x, y)^T$$
$$\overrightarrow{b} = \begin{pmatrix} (d_1^2 - d_3^2) + (x_3^2 - x_1^2) + (y_3^2 - y_1^2) + (z_3^2 - z_1^2) \\ (d_2^2 - d_3^2) + (x_3^2 - x_2^2) + (y_3^2 - y_2^2) + (z_3^2 - z_2^2) \end{pmatrix}$$

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

For the computation of the x and y coordinates, only simple arithmetic operations are needed such as addition, subtraction, and multiplication. Those are very basic (and fast) operations, available on low cost micro-controllers. 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 Equation 2, which, however, requires a square root function for the used micro-controller.

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

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

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 $m = \binom{n}{3} = \frac{n!}{3! \cdot (n-3)!}$ disjunct subsets of three pairs can be computed. The number of possible subsets increases significantly with the number of reference points. In terms of scalability it is not feasible to compute all m subsets and to evaluate them.

Casas and co-workers [16] 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 [16]:

$$m = \frac{log(P_m)}{log(1 - (1 - P_{mf})^3)} \tag{6}$$

Thus, 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 to 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. Afterwards, it can be placed in a sorted 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) non-regular tetrahedron (3 reference points plus the unknown point). The base area of the figure is a triangle.

Usually, this can not be computed very fast because square root or trigonometric functions would be needed. 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. According to (3), the base area is parallel to the x–y plane, so the cross product only contains a z component (Equation 7). This length can therefore be computed very fast, only summation, subtraction, and multiplication methods are needed.

$$\overrightarrow{a} \times \overrightarrow{b} = \begin{pmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ a_1b_2 - a_2b_1 \end{pmatrix} \tag{7}$$

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 the mean of all positions. However, outliers would significantly influence the result. Casas et al. [16] used an approach where a squared residual vector between all measured and all theoretical distances for each subset is 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.

In a second step, we incorporated prior knowledge into the position estimator. Casas method [16] 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 in a similar way like an extended Kalman filter. In the first step, we predict the current position $\overrightarrow{x_p}(k+1)$ using a state vector:

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

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

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. We generate the new position $\overrightarrow{x}(k+1)$ by selecting the nearest computed position to the predicted position out of the set $X(k+1)$.

The more time has elapsed since the last computation in relation to the last interval, the less reliable the prediction gets. The correction function $\kappa()$ in Equation 8 has been designed for compensating this effect. It is a function of r (Equation 9), which denotes the ratio of two time intervals. $\kappa()$ is a simple function that returns 1 for values between 0 and 1. For greater values, the output slowly decreases 0. Figure 2 illustrates the prediction vector and the growing space of the position acceptance. As shown on the left side, the prediction vector grows 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 means that the last localization interval (i.e., the time between two accepted positions) was shorter than the elapsed time since the last position was accepted. Now, $\kappa()$ is being decreased because at this time a proper prediction based on the movement during the last interval can not be guaranteed.

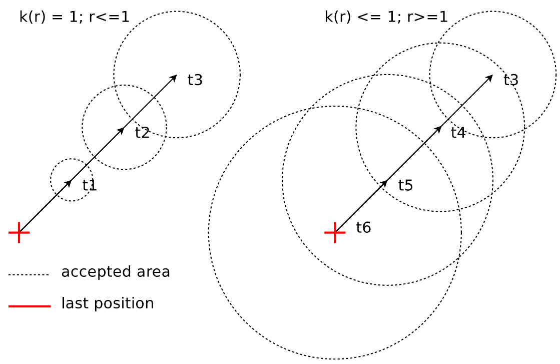

Fig. 2. Position prediction

We achieved very good and fast results using this estimation technique. However, it can happen that a wrong position is accepted. For example, if all taken measurements are wrong and, therefore, all possible positions are as well incorrect. In this case, the position estimator takes the nearest of these wrong position if it is within the accepted area. So, invalid state and 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 calculated, the state will become accurate again within a few cycles.

IV. Localization Performance

Scalability is one of the biggest issues in the context of sensor networks. In order to proof our localization algorithm works even on resource constricted embedded systems, we implemented the system and evaluated it in a lab scenario. In particular, we used the SunSpot sensor node platform [17] running JavaME as the host operating system. We first estimated the computational performance of the localization algorithm. In the next section, we discuss the applicability for real-time localization of our flying quadrocopter.

One of the key issues is the creation of the subsets. Figure 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 (depicted in red in Figure 3) can be given. The limitation of subsets implicitly restricts the position vector calculation time to an upper boundary, too. Thus, not every possible position needs to be calculated: Only positions from subsets that meet a certain threshold in the qualification are being considered.

In Figure 4, all the tested timings of the position estimation algorithms are depicted. As mentioned before, the residuum based method [16] (blue) scales approximately linearly with the number of reference points. Similarly, the "only point" based algorithm (red), which recursively generates the mean of all positions and removes the position that is most distant

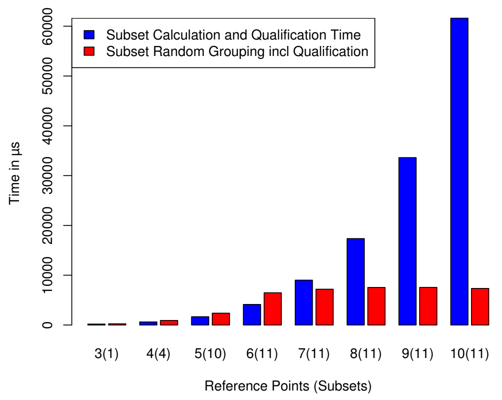

Fig. 3. Subset calculation

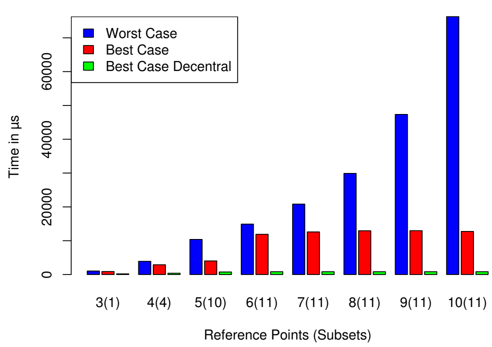

Fig. 5. Total localization time

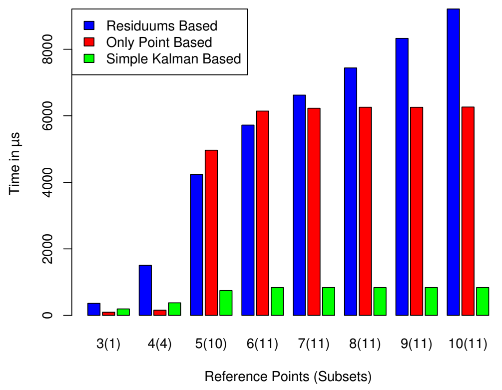

Fig. 4. Position estimation

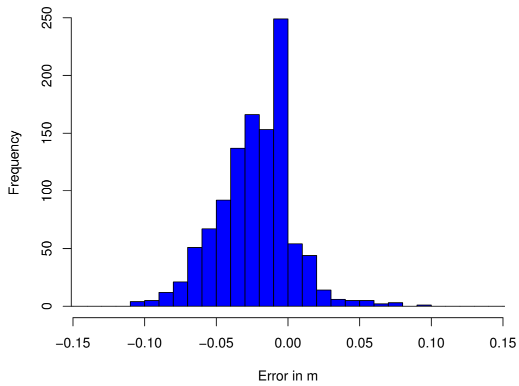

Fig. 6. Localization accuracy: the quadrocopter hovers over the ground, the height error is plotted

to the mean, does not scale well, the computation costs are too high, and the demands for accuracy cannot be met. The Kalman based predictive method (green) gives accurate and quick results.

Finally, Figure 5 shows the total computation time. The worst case scenario (blue) 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 (red), only techniques with a bounded computational time are used. So an upper boundary for the localization algorithm can be given independent of the number of used reference points. This is important to fulfill the real-time specification. The best case decentral scenario (green) 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 real-time control of a quadrocopter. We also discuss the practical implementation of the developed algorithms. Despite the classical master-slave topology, we decided for a hybrid measurement architecture. Whether a device is master (transmitter) or slave (receiver) is completely hardware independent and can be controlled on application level. The detection field of the system is designed to be a hemisphere. Thus, the reference points on the floor can not only detect the flying object but also each other (this architecture is depicted in Figure 1). This way, it is possible to span up the grid automatically by attaching the reference points on top of mobile robots. Another advantage of a flying active beacon, as mentioned before, 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.

In order to measure the localization accuracy, we arranged one reference point in each corner of a square, so in total four reference points are used. The length of the edges was $2\,\mathrm{m}$. The object hovers randomly in a square of about $3\,\mathrm{m}$ of edge length and at an altitude of $0.5–2.5\,\mathrm{m}$ over the reference square. Figure 6 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 $\pm 10\,\mathrm{cm}$ can be achieved with a confidence of $98\,\%$.

For the measurements shown in Figure 7, we placed the

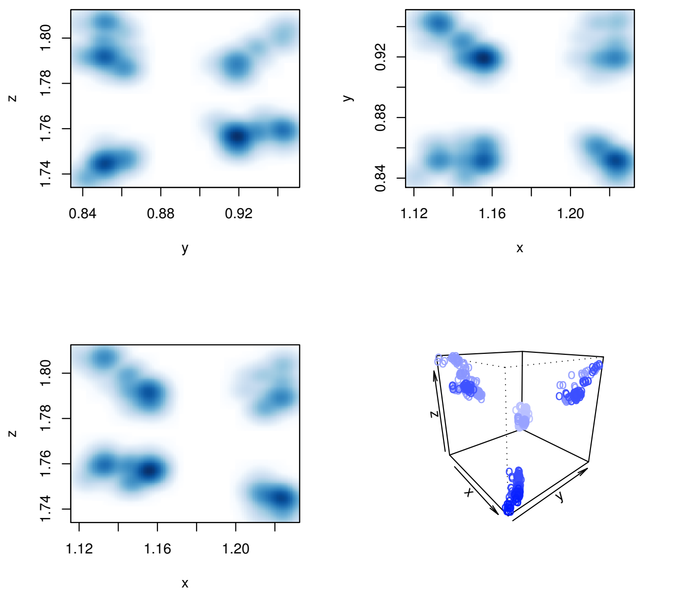

Fig. 7. Localization accuracy: the quadrocopter is fixed in a position, the measured x, y, z coordinates are plotted

four-rotor robot at an arbitrary but fixed position over the detection field. It can be seen 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 $\pm 2\,\mathrm{cm}$. The estimated position is normally confined to one of those regions. But 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 too far away. Secondly, 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 $\pm 3\,\mathrm{cm}$. With increasing deployment accuracy of the reference points, the resulting regions merge into a single one.

VI. Conclusion

We 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 are not possible in this case. Thus, a real-time localization is needed that must also take weight constraints into account.

Considering these requirements, we developed an algorithmic procedure that advances the state of the art in indoor localization by being able to perform real-time localization based on possibly error-prone distance measurements. 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 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 lab. The results demonstrate the feasibility of the solution. We consider our ultrasound lateration technique a necessary step for completely autonomous operation of flying robots in indoor environments.

References

[1] A. Smith, H. Balakrishnan, M. Goraczko, and N. Priyantha, "Tracking Moving Devices with the Cricket Location System," in *2nd International Conference on Mobile Systems, Applications, and Services (MobiSys 2004)*. Boston, MA: ACM, June 2002, pp. 190–202.

[2] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The Cricket Location-Support System," in *6th ACM International Conference on Mobile Computing and Networking (ACM MobiCom 2000)*. Boston, MA: ACM, August 2000, pp. 32–43.

[3] J. Eckert, F. Dressler, and R. German, "An Indoor Localization Framework for Four-rotor Flying Robots Using Low-power Sensor Nodes," University of Erlangen, Dept. of Computer Science 7, Technical Report 02/09, May 2009.

[4] R. Want, A. Hopper, and V. Gibbons, "The Active Badge Location System," *ACM Transactions on Information Systems*, vol. 10, no. 1, pp. 91–102, January 1992.

[5] C. Durieu, H. Clergeot, and F. Monteil, "Localization of a Mobile Robot with Beacons Taking Erroneous Data into Account," in *IEEE International Conference on Robotics and Automation*, Scottsdale, May 1989, pp. 1062–1068.

[6] J. Leonard and H. Durrant-Whyte, "Mobile robot localization by tracking geometric beacons," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 376–382, June 1999.

[7] R. Salomon, M. Schneider, and D. Wehden, "Low-Cost Optical Indoor Localization System for Mobile Objects without Image Processing," in *IEEE Conference on Emerging Technologies and Factory Automation (ETFA 2006)*, Praha, September 2006, pp. 32–43.

[8] P. Bahl and V. N. Padmanabhan, "RADAR: An In-Building RF-based User Location and Tracking System," in *19th IEEE Conference on Computer Communications (IEEE INFOCOM 2000)*, Tel-Aviv, Israel, March 2000.

[9] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low-cost outdoor localization for very small devices," in *IEEE Personal Communications*, October 2000, pp. 28–34.

[10] A. Mandal, C. Lopes, T. Givargis, A. Haghighat, R. Jurdak, and P. Baldi, "Beep: 3D indoor positioning using audible sound," in *2nd IEEE Consumer Communications and Networking Conference (IEEE CCNC 2005)*, Las Vegas, January 2005, pp. 348–353.

[11] A. Mahajan and M. Walworth, "3D Position Sensing Using the Differences in the Time-of-Flights From a Wave Source to Various Receivers," *IEEE Transactions on Robotics and Automation*, vol. 17, pp. 91–94, February 2001.

[12] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster, "The anatomy of a context-aware application," *ACM/Springer Wireless Networks*, vol. 8, pp. 187–197, March 2002.

[13] W. Foy, "Position-Location Solutions by Taylor-Series Estimation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-12, pp. 187–194, March 1976.

[14] L. Kleeman, "Optimal estimation of position and heading for mobile robots usingultrasonic beacons and dead-reckoning," in *IEEE International Conference on Robotics and Automation*, vol. 3, Nice, May 1992, pp. 2582–2587.

[15] 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," *Sensors and Actuators: A. Physical*, vol. 75, pp. 131–138, June 1999.

[16] R. Casas, A. Marco, J. Guerrero, and J. Falco, "Robust Estimator for Non-Line-of-Sight Error Mitigation in Indoor Localization," *EURASIP Journal on Applied Signal Processing*, vol. 2006, pp. 1–8, 2006.

[17] R. Smith, "SPOTWorld and the Sun SPOT," in *6th International Conference on Information Processing in Sensor Networks*, Cambridge, April 2007, pp. 565–566.