A-GNSS
A Different Approach

ISABELLE KRAEMER AND BERND EISSFELLER
UNIVERSITY FAF MUNICH

More and more mobile devices are integrating GNSS technology – and this increases the pressure to improve positioning capabilities in difficult environments. In this column, the authors introduce a novel use of peer-to-peer transfer of position information that combines GNSS and dead-reckoning techniques with Bluetooth communications to assist positioning indoors.

When the GPS program was established, military services were undoubtedly targeted as its primary user group, focusing on outdoor operations and offering the capability of continuous tracking. In addition we have heard several times that GPS was planned as a dual-use system from its very beginning.

Given the nature of its design, however, GPS clearly was not primarily intended for use in urban canyons or indoors where its weak, line-of-sight signal often cannot reach. Nevertheless, the demand for portable devices with an integrated GPS chip continues to increase steadily and in consequence also the desire to navigate not only on highways by car but also in cities with tall buildings and especially within buildings.

According to one estimate, almost 30 percent of all mobile phones that will be sold between 2009 and 2011 have an integrated GPS chip. The increasing rate of mobiles sold with integrated GPS chips has created a distinct change in the use of civilian navigation — by markedly expanding its use pedestrians as well as motorists, indoors as well as on the highways.

Various methods have been developed to make navigation in environments with weak signal coverage possible. Unfortunately they either require an accurately surveyed area (e.g., WLAN "fingerprint"), additional equipment (e.g., gravitational sensors), or additional data transmission (assisted-GNSS or A-GNSS).

The latter principle — A-GNSS — is well known: by sending assistance data from a server via the radio network a device is able to perform a "hot start" and acquire a position fix within seconds rather than the lengthy time-to-fix-first usually required with a cold start under normal situations. Moreover, A-GNSS also helps in increasing the sensitivity of the receiver and in acquiring signals within buildings.

The improved results from A-GNSS are already substantial, but considerable research indicates that the performance within buildings is still not completely satisfying. Other drawbacks include the requirement for users to buy an A-GNSS-capable device and to pay for the data transmission. Moreover, when many such devices use A-GNSS service in one area at the same time, the server calculating and providing the aiding data might soon be overstrained.

Some approaches to indoor navigation do not rely on GNSS, but they often require an already well-surveyed area with infrastructure, for example, WLAN access points or RFID readers.

This column will introduce an approach that avoids these drawbacks and uses short-range peer-to-peer communication to enhance the estimation of a position. Portable devices calculate their position by dead reckoning when no satellite signals are available. To keep the approach simple only a compass and a step detector are used. When two of these devices can communicate their results, they can also try to improve their positions by using a Kalman filter.

Accordingly, the devices will use

GNSS to calculate a position whenever possible (e.g., near doors, windows, or reference stations) and otherwise use dead reckoning to estimate their position based on stride length estimation and heading from a magnetometer.

When two or more devices come within reach (i.e., communications range) of each other, they compare their individually estimated positions and calculate a corrected one based on weighted average.

This article will first discuss Bluetooth, the communications technology of choice for this approach. Then we will briefly explain the operation of the Kalman filter for this application. Finally, we introduce our A-GNSS method accompanied by a simulation model and results of test simulations.

This new approach addresses two shortcomings of existing indoor navigation technologies:

- The user does not have to pay additional fees for any data transmission or new devices.
- The need for a centralized infrastructure network can be avoided in favor of small local ad-hoc networks.

Bluetooth Ad Hoc Network

A wireless ad hoc network establishes connections between various devices without relying on external hardwired infrastructures. In general, an ad hoc network only lasts for a short time during which one device can transmit data to another and vice versa. The IEEE 802.15.1 communications technology better known as Bluetooth is an industrial standard that provides the specifications to establish such connections.

The Bluetooth protocol stack is divided into the so-called *core specification* and the *profile specification*. The former describes the protocols of the physical layer and the data link layer while the latter specification contains several protocols and functions to fit Bluetooth to different applications.

The data transmission takes place in the 2.4 GHz industrial, scientific, medical (ISM) frequency band. There are 79 channels with a distance of one megahertz between each other. Bluetooth uses FHSS (frequency hopping spread spectrum) with 1,600 hops per second. To distinguish various so-called *piconets* from each other, a frequency hopping code division multiple access (FH-CDMA) technique is used.

A piconet consists of up to 8 devices that can actively communicate with each other and up to 247 parked devices — temporarily inactive units not taking part in any communication but synchronized to the piconet's hopping sequence. One of the active devices serves as the master while all other devices are slaves.

The master only defines the hopping sequence to which all the slaves have to synchronize. Any device that is able to use Bluetooth can become a master. A master simply starts the conversation in one piconet but may also be a slave in another. Such overlapping piconets are called *scatternets* (see **Figure 1**).

To start a piconet as a master or to look for an existing net, the device begins in a state called *inquiry*. If the device seeks to be the master of a piconet, it broadcasts an inquiry access code (IAC) on all 32 standardized carrier frequencies. Conversely, the device searches periodically for IAC messages when becoming a slave.

Whether a master or slave, as soon as another device has been found, a Bluetooth-enabled device changes its state to *page* and, after the synchronization, to the state *connected*. To exchange data the devices switch to *transmit* mode.

Figure 2 displays the states and the transitions between them. Those shown on the bottom are for power-saving only.

To communicate with each other Bluetooth uses FH/TDD (a combination of frequency hopping and time division duplex). One time slot has a duration of 625 microseconds in which either the master or one of the slaves is allowed to send. After the time slot expires, the frequency is changed according to the hopping sequence calculated by the master. Up to 343 bytes of data can be transmitted per packet.

Some efforts have already been made to use Bluetooth for indoor navigation. For example, the work presented in the article by S. Feldmann et alia, which is listed in the Additional Resources section near the end of this article, has proposed a system in which the position of a mobile device is calculated by measuring the amount of delay in transmissions from at least three access points and the mobile device. K.Thapa and S. Case suggest a similar approach in their article (Additional Resources).

Both methods measure the signal strength by requesting the *Received Signal Strength Indicator (RSSI)*. This value expresses the signal strength between the local device and the connected device by relating it to the "Golden Receive Power Range," which determines the desired received signal power that results in minimal error. The maximum of this range for all devices is -56 dBm.

The RSSI range of an individual device is defined by an offset of 6 dB of the receiver sensitivity (the lower threshold) and the upper threshold adding 20 dB to the lower threshold. The RSSI returns a negative value if the RSSI is too low, zero if the RSSI is within the "Golden Range," and a positive value if the RSSI is too high. This makes it hard to calculate the position by triangulation. Moreover, as the Feldmann et alia

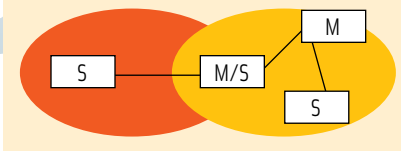

FIGURE 1 **Two piconets forming a scatternet. The device in the middle is master (M) in one of the piconets and a slave (S) in the second piconet. The master dictates the hopping sequence.**

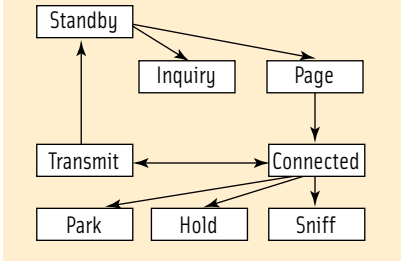

FIGURE 2 **The state diagram of a Bluetooth device. The three states at the bottom are taken when the device is in power-saving mode.**

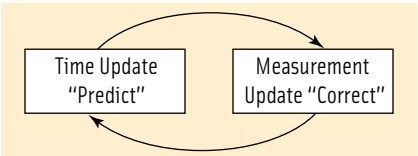

FIGURE 3 The Kalman Filter cycle. In the prediction phase the current state is estimated ahead in time; the correction phase adjusts the estimations done in the predictions phase.

article explains, one can only use RSSI values that are positive to do the triangulation and the distance between the device and the sender must not exceed eight meters.

To avoid these drawbacks, Bluetooth is only used here as a cheap and reliable communication protocol between two or more devices. The most significant advantages of Bluetooth compared to any other data transmission protocols are its wide use, high acceptance rate, and its existing integration in almost every kind of portable device. Nowadays, even notebook computers have a Bluetooth unit.

Furthermore, new protocols for Bluetooth are frequently added. Since the first appearance of this standard, new applications and protocols have been widely developed to expand the functionality and to adapt it to new applications. This provides the opportunity and makes it easy to define and implement a protocol that fits the needs of the peer-to-peer navigation model that we will present in this article.

For our A-GNSS method, we don't need to determine the exact distance between two devices; knowing that another device is relatively near to the other is enough. This serves our purpose due to the small cell size of Bluetooth networks.

All Bluetooth devices are grouped in three power classes according to their maximum transmission power, which limits their coverage area. The transmission power ranges from 100 milliwatts for Class 1 devices that are able to cover an area of 100 to 150 meters to 1 milliwatt for Class 3 devices with a coverage of less than 10 meters.

Most Bluetooth devices are Class 2 devices that have a maximum range to another device of 50 meters but mostly do not exceed 10 meters within buildings. This guarantees that devices which use a Kalman filter to improve their position mutually are not too far away from each other and will not degrade the results unduly.

Dead Reckoning and Kalman Filter

The principle of dead reckoning using step detection is very easy to explain. One starts with a more or less precise knowledge of a position and calculates for each step the new position based on new measurements.

This approach assumes that the portable device has a measuring sensor such as a compass in combination with a step-detection unit. For simulation purposes, the length of one stride is always assumed to be of 0.7 meter, which can be considered to be a realistic value based on substantial investigation by numerous researchers.

The coordinates of the dead reckoning position are calculated as follows:

$$X_k = X_{k-1} + s_k \cos(\psi_k) \quad (1)$$

$$Y_k = Y_{k-1} + s_k \sin(\psi_k) \quad (2)$$

where X_{k-1} and Y_{k-1} indicate the coordinates of the step k-1, s_k provides the length of the k-th step, and ψ_k relates to the heading.

Of course the measurement of the heading is error-prone due to the precision of the compass in the portable device and the fact that users don't always have a stride length of 0.7 meters. Also, dead-reckoning sensors tend to accumulate more errors the further one travels. As a consequence, the estimated position will differ more and more from the true position over the time.

To avoid a deviation too high to be useful, a correction of the estimated position has to be made and the error calculated. This is done by Kalman filtering. The functionality of a Kalman filter is well-known: It implements a predictor-corrector type estimator optimized to decrease the estimated error covariance. The filter cycles between the two states of prediction and correction, as shown in Figure 3.

In the prediction phase the current state and the error covariance estimates must be calculated based on the estimates of the earlier state. This is done by:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k \quad (3)$$

$$P_k^- = AP_{k-1}A^T + Q \quad (4)$$

The n x n-Matrix A relates the state at the previous step k-1 to the state of the current step k. The n x l-Matrix B relates the optional control input u to state x. The value of $\hat{x}_k^-$ is the estimated current state before any corrections are made. The value of P_k^- represents the estimation of the error covariance before the correction. Q is the process noise covariance.

For the correction the following equations are relevant:

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \quad (5)$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \quad (6)$$

$$P_k = (I - K_k H)P_k^- \quad (7)$$

The first value that must be obtained in the correction phase is the so-called *Kalman gain* K_k. In equation (6) the expression $(z_k - H\hat{x}_k^-)$ is called the measurement *innovation* or *residual*. It reflects the discrepancy between the predicted measurement $H\hat{x}_k^-$ and the actual measurement z_k. If the residual is zero, the predicted measurement and the true measurement are consistent.

This equation computes the state $\hat{x}_k$ after the measurement as a linear combination of the estimate before the measurement $\hat{x}_k^-$ and a weighted difference of the actual measurement z_k and the predicted measurement $H\hat{x}_k^-$. Equation (7) updates the error covariance.

The design of this approach follows the rules and equations of the Kalman filter. To verify the functionality, we developed a simulation model based on Matlab. The next section explains the design of this simulation in detail and relates the formulas of the Kalman filter that we have introduced in this section to the equations used in the simulation.

Simulation Model

As mentioned before, the main intention of our approach is to enable users

to navigate in an indoor area independent of A-GNSS or any pre-installed infrastructure. The users' positions are estimated by dead reckoning, which implies that each user has a rough idea of its position when starting the simulation, which we developed on a PC with the help of a high-level language and interactive environment software package.

To display the effects of the Kalman filter, the simulation must maintain two positions and the error variance of the position. One position is the true position, the other a position that is estimated by the user's device based on the compass measurements and the assumption that the stride length is 0.7 meter.

The indoor area is simulated as a flat, square plane with a dimension of, for example, 100 × 200 points where one point corresponds to 1 meter (see **Figure 4**). In this area the users, represented by the black stars, move around. The black star indicates the *true* position. The red stars on the left and on the right side indicate *reference* positions. These are areas where a true position can be obtained using satellite positioning.

Speaking in terms of the Kalman filter, this is where the correction of the previously measured position takes place. A blue line indicates the distance between true position and estimated position plus the known error. The circle around the estimated position is the error variance.

The walkers (carrying the positioning devices) are randomly dispersed in the simulation area with their heading chosen arbitrarily. A simulated walker moves in a straight line until and unless it meets a wall. If it hits the limits of the indoor area (a "wall") it turns around at a 90-degree angle and again moves in a straight line.

In each simulation step every walker must execute four operations:

1. calculation of the true position of the walker
2. calculation of the estimated position of the walker
3. testing if the walker is near a reference position
4. testing if the walker is near another walker.

For simplicity we assume that the walker moves in each simulation step and never stops. That is why a step detector is not considered in the simulation. Therefore, the walker's new true position has to be calculated for each time step.

The formula to calculate the new position is exactly the same as (1) and (2) but s_k (the stride length) is calculated as follows:

trueStride = 0.9 + randn() * dStrideAcc;

Of course, the test calculates whether the walker still resides inside the simulation area. If not, the heading is changed and the walker changes direction to stay within the area. Furthermore, the simulation model assumes that an average adult human walks with a stride length of 0.7 meter. To simulate the differences for each walker this static value is combined with a so-called stride accuracy (dStrideAcc), which is randomized. The stride length has a standard deviation of 0.1 meter.

This seems a very small value for a deviation factor, but with a larger deviation value we would have to measure not only the heading of a walker but also the stride length — which would require us to take another measurement unit into consideration.

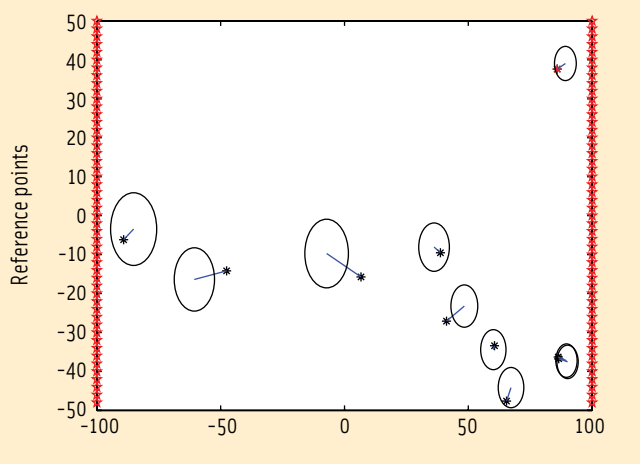

FIGURE 4 The simulated indoor area. The red stars on the left and the right side mark the reference positions. The black stars are the walkers' true positions. The blue lines indicate the distance from true to estimated position. The ellipsoids display the error variance of the estimated position. There is one walker (top right), which is marked with a red star. This walker is randomly chosen and his values displayed in an extra plot.

Many relatively cheap and small devices that would serve this purpose are already on the market; so, conceivably we could factor a larger deviation into the simulation. This would even improve the position estimation when using only dead reckoning.

The next step in the simulation is to update the estimated position of each walker. Before the simulation begins, each walker knows its position; so, at that moment estimated and true position are consistent with each other.

To calculate the estimated position formulas (1) and (2) are used. The difference to the calculation of the true position is that here a static stride length of 0.7 meter is assumed, and the true heading of the walker is distorted by the compass accuracy:

headEst = headTrue + randn( ) * nCompassAcc;

The standard deviation of the compass is 15 degrees and, of course, also randomized. All formulas to calculate true and estimated position are listed below:

True Position

$$x\_true_k = x\_true_{k-1} + s\_true_k \cos(\psi\_true_k) \quad (8)$$

$$y\_true_k = y\_true_{k-1} + s\_true_k \sin(\psi\_true_k) \quad (9)$$

Estimated Position

$$x\_est_k = x\_est_{k-1} + s\_est_k \cos(\psi\_est_k) \quad (10)$$

$$y\_est_k = y\_est_{k-1} + s\_est_k \sin(\psi\_est_k) \quad (11)$$

Of course, when calculating the estimated position the error variance must also be determined. This is done using the following formula:

$$P_k = P_{k-1} + (s\_est_k \cdot compAcc)^2 + strideAcc^2 \quad (12)$$

Formulas (10) to (12) describe all the calculations that

have to be done during the prediction of the Kalman filter. The walker knows its estimated position and the estimated heading due to the compass and assumes a static stride length of 0.7 meter. The walker also knows that the compass might be wrong by about 15 degrees and the assumed stride accuracy by 0.1 m. The true position, the true heading, and the true stride length are only known to the simulation.

After calculating true and estimated position, the simulation determines whether the walker is near a reference position. Within a radius of two meters around the reference position, the test assumes that a walker can estimate its true position or, to express it in different terms, to correct its estimated position.

In the simulation introduced here reference positions are only available at both ends of the indoor area indicating windows or doors where a position based on GNSS can be determined. Conceivably we might also add reference positions within a building, but this runs contrary to our approach's drivers of having no pre-installed infrastructure at all. If one did use such reference positions inside a building, their coordinates should be accurately surveyed and transmitted to the walker via a wireless communication protocol, e.g., Bluetooth. In addition, the distance between the reference position transmitter and walker must be very small to achieve a beneficial correction.

In the simulation the walker simply calculates its error, which means the distance between the reference position and its estimated position. And, of course, the error variance is updated here. In contrast to the common Kalman filter, the Kalman gain and the true state — see equations (5) and (6) — are not calculated.

For the simulation it is sufficient to define the residual between estimated measurement and true measurement as x- and y-values. This residual also is known to the walker and memorized in its device (x_{err} and y_{err}). The new error variance is the squared maximum distance a walker is allowed to have to a reference position. In this case the maximum distance is two meters; so, the error variance is four meters.

A final test has to be made before one simulation step is finished, namely, to control whether a walker resides in the sending area of another walker. This simulation assumes that a walker only has a transmission range of 2 meters. Two walkers that reside within this radius are allowed to correct their estimated positions.

Here is how the procedure works: Both walkers calculate a new x- and y-value based on their estimated positions and their residuals:

$$x_i = x\_err_i + x\_est_i \qquad (13)$$
$$y_i = y\_err_i + y\_est_i$$

In each case the weighted averages of these new x- and y-value are computed, based on:

$$x = \frac{\frac{1}{var_1}x_1 + \frac{1}{var_2}x_2}{\frac{1}{var_1} + \frac{1}{var_2}} \qquad (14)$$

$$y = \frac{\frac{1}{var_1}y_1 + \frac{1}{var_2}y_2}{\frac{1}{var_1} + \frac{1}{var_2}}$$

where var_i is the error variance and x_i is the x-value calculated from (13). The y-value is computed in the same way.

Based on these two new coordinates the residuals of the walkers are calculated and stored as x_{err_i} and y_{err_i}. According to the probability calculation the new error variance of each walker is updated by:

$$var_i = \frac{1}{\frac{1}{var_1} + \frac{1}{var_2}} \qquad (15)$$

The whole algorithm as implemented in Matlab can be found in **Algorithm 1**. The estimated position, the variance, and the residuals are stored in vectors with an element for each walker.

As given by the definition of the Kalman filter, it is only possible to process one correction for each walker in every simulation step. This means that a walker who already updated its position at a reference position is not allowed to make a second correction with another walker in the same simulation step. Also, a walker cannot compute corrections with more than one other walker in the same time step.

```
xn = vecXDr(n)+vecXErr(n);
yn = vecYDr(n)+vecYErr(n);
xm = vecXDr(m)+vecXErr(m);
ym = vecYDr(m)+vecYErr(m);

%calculate then an estimated position
for both walkers based on weighted average
x = ( xn/vecErrVar( n ) + xm/vecErrVar( m
)) /(1/vecErrVar( n ) + 1/vecErrVar( m ) );

y = ( yn/vecErrVar( n ) + ym/vecErrVar( m
)) /(1/vecErrVar( n ) + 1/vecErrVar( m ) );

%calculate the error based on the
calculated new position
vecXErr(n) = x - vecXDr(n);
vecYErr(n) = y - vecYDr(n);

%error variance is also calculated based on
%weighted arithmetic average
vecErrVar( n ) = 1/(1/vecErrVar( n ) +
1/vecErrVar( m ) );

vecErrVar( m ) = 1/(1/vecErrVar( n ) +
1/vecErrVar( m ) );
```

ALGORITHM 1 **The algorithm with which to calculate the correction of two walkers that come into each other's zone of communication**

Simulation results have shown that concentrating on the closest neighbor results in the highest improvements in accuracy. Indeed, feeding the position of various walkers in the vicinity into the Kalman filter does not further improve the performance of this approach; however, in this case the computational load significantly increases and the communication link also faces a higher burden of traffic.

The next section presents the results of the simulation. The distance between true position and estimated position will be plotted for each walker, and any improvement achieved by the correction of other walkers is indicated.

Simulation Results

To evaluate the peer-to-peer model we measured the distance between the true position and the estimated position plus the known error of each walker

and averaged these for each simulation step. At the end of the simulation this value is also averaged for all walkers in order to generate a single value that can be compared.

Additionally, for each simulation one walker is randomly chosen to watch the behavior when encountering another walker, coming to a reference position or reaching the limits of the indoor area. Three parameters are varied for each simulation: number of walkers, size of the indoor area, correction with walkers and reference positions (correction_true), or only with reference positions (correction_false).

The simulation only allows a walker to correct its position when it comes within a range of two meters of a reference position. The two meters are an estimated value. We assume that a walker could only calculate a useful position by GNSS when it is no more than two meters away from a window or door area. The walker incorporates the first reference position at which the distance between itself and the reference point is smaller than two meters.

In real life, the walker's device would estimate its position by dead reckoning until it realizes that a GNSS signal can be acquired. Then it corrects the estimated position with the position calculated using GNSS signals. In the simulation, we used two modes to point out that the accuracy of the position improves when the walker is allowed to correct its estimation with other walkers' position estimation AND reference points (correction_true) instead of only correcting with reference points (correction_false).

The variation of the number of the walkers for each simulation follows the scale 100, 200, 300, and 500 (only for the indoor area of greater size). The size of the indoor area ranges from 100 x 200 meters to 400 x 600 meters. Every simulation instance is processed with the correction varying the involvement of other walkers. So, each simulation (characterized by size of the indoor area and number of walkers) is done twice: The first time the walker is only allowed to correct its position with reference points the second time he is allowed to correct with reference points *and* other walkers.

The first simulations that we processed required only a distance to other walkers below five meters. However, this maximum limit turned out to be too large to calculate a reasonable correction consistently. This is why the maximum distance between two walkers providing peer-to-peer aiding must not be more than two meters. Another requirement is that only the two walkers having the smallest distance between themselves of all those within two meters should correct each other's position.

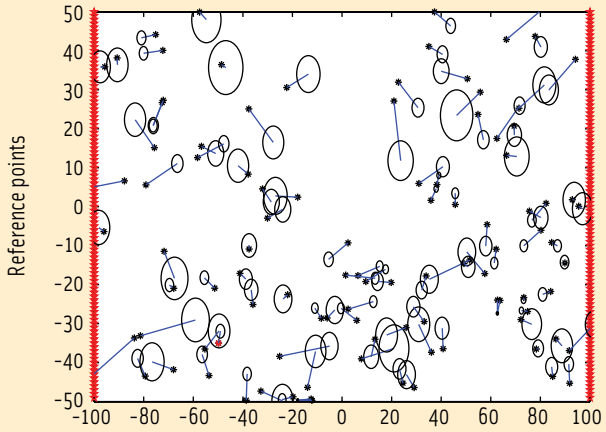

FIGURE 5 The indoor area for the first part of the simulation. The area has dimensions of 100 x 200 meters.

The parameters of the first simulations are: indoor area, 100 x 200; reference points — 200, 100 on each side of the indoor area (see Figure 5).

The size of errors for both kinds of simulation — correction_true (another walker and reference point) and correction_false (only a reference point) — in the *x*- and *y*-direction do not differ strongly. For the simulation with correction_false, the average difference in *x*-direction is 7 meters and in *y*-direction 3.7 meters, while for the simulation using correction_true the distance in *x*-direction is 4.6 meters and 3.3 meters in the *y*-direction.

Figure 6a displays the distance to the true position of the arbitrarily chosen walker. Green colored areas indicate where the walker encountered another walker and corrected its position; the blue areas are points where the walker corrected its position with the help of a reference position.

In Figure 6b one can see the results of this first simulation. As the opportunities for obtaining useful aiding information are very low with only 100 walkers, the accuracy sometimes even becomes worse when correcting by use of another walker's position. As mentioned earlier, among all walkers that are within its communications range, a walker chooses the one that is nearest. When too few walkers are within range, this choice is obviously limited.

With more walkers, however, the average position accuracy increases as expected. In our second scenario, we doubled the number of walkers: instead of 100 walkers, 200 now stroll about the indoor area.

We measured the positioning accuracy for the simulation without a correction with other walkers, although the accuracy should not change at all. This is, indeed, true. The average error in the *x*-direction for this correction_false calculation still is around 7 meters and the error in the *y*-direction increases a little to 4.7 meters.

More significantly, as expected the position accuracy improves for the 200-walker simulation using correction_true — incorporating the position information of other walkers as well as the reference positions. In the *x*- and *y*-direction, the average errors are 3.6 and 3.1 meters, respectively.

This tendency still holds when increasing the number of walkers to 300. The errors in *x*- and *y*-directions decreases to 3.1 and 2.6 meters, respectively, while the accuracy of the correction_false calculations stays nearly the same. In Figure 7 one can see that the correction with another walker really is an improvement, while this was not always true when only 100 walkers were present.

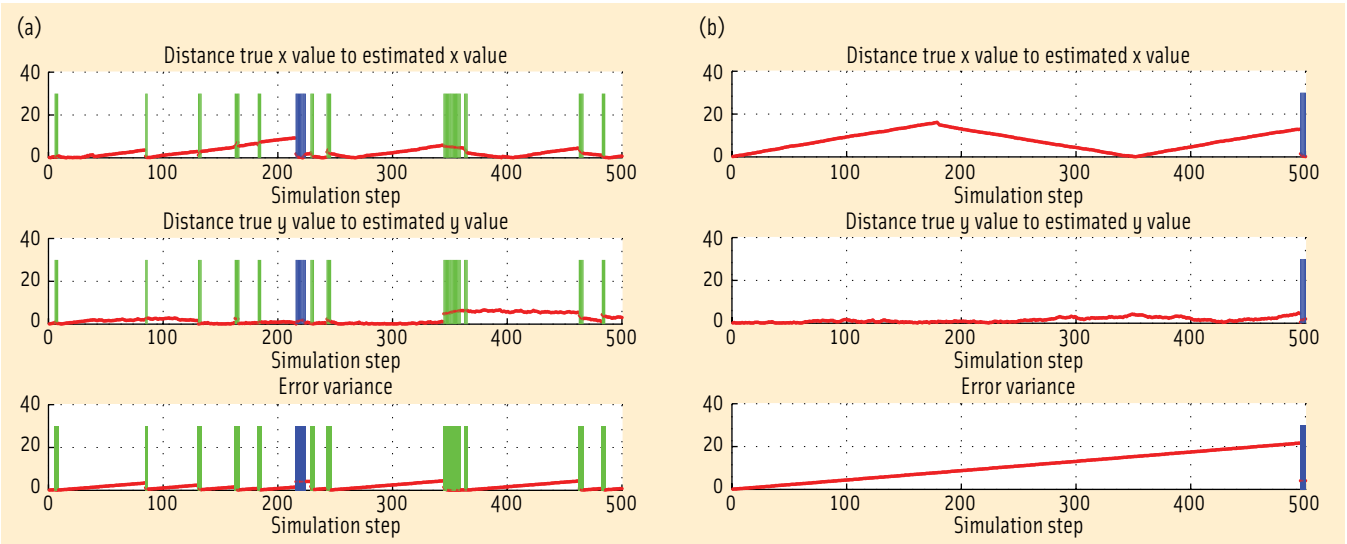

FIGURE 6 Data of the arbitrarily chosen walker. Both diagrams show the absolute difference of the estimated to the true position in x- and y-direction (vertical scale in meters). Diagram (a) is the result of a simulation with correction by other walkers and reference positions (green areas) while in (b) the correction was only done with reference positions (blue areas). Parameters of the simulation: 100 walkers, indoor area 100 x 200 meters with 200 reference positions.

The overall deviation from the true position for all the walkers can be found in **Figure 8**. Figure 8a displays the absolute average distance to the true position in *x*- and *y*-direction for each walker. Figure 8b and c show the frequency distribution of the distance from true position in *x*- and *y*-direction with an increasing number of walkers present.

The figure data reflect the fact that correcting a position with the aid of another user does not always produce an improvement in accuracy, but the quality of the correction becomes better the more walkers are in the area. As one can easily see, the best results evolve from the simulation with 300 walkers.

This improvement stems from two things: a walker has more choice for being aided when more walkers are nearby so that it can choose one that is really close, and a walker can correct more often. The figure only shows that a selected walker met another walker, but the simulation's algorithm ensures that this is always the nearest one.

As to be expected, the peer-to-peer model works better if more walkers are available to correct each other's estimated position.

For the second simulation, we increased the size of the indoor area to 400 x 600 meters, while the number of reference positions remained the same (see **Figure 9**). For the uncorrected case (a correction from the reference position but not from other walkers) the average position errors are about 12.5 meters in the *x*-direction and 10.8 meters in the *y*-direction. This figure, of course, does not change when increasing the number of walkers.

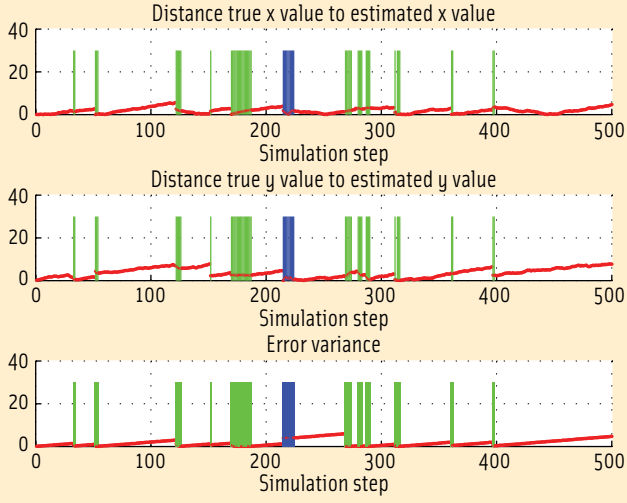

FIGURE 7 Instead of only 100 walkers, this simulation assumes that 300 were in the indoor area. In most cases, the corrections provided by other walkers really do improve position accuracy.

However, accuracy indeed improves for the corrected case, as can be seen in **Figure 10.** The first simulation also produces very high errors in *x*- and y-direction (9.8 and 10.3 meters, respectively) when only 100 walkers are in the indoor area but improves to 8.5 and 8.2 meters for 300 walkers and even to 8.1 and 7.1 meters with 500 walkers present.

As to be expected, the peer-to-peer model works better if more walkers are available to correct each other's estimated position. When the indoor area in relation to the reference position is very small as in the first simulation, the correction of other walkers might even degrade the estimated position, depending on how long the other walker has moved without a correction (accumulated error). In this case, it may be better to

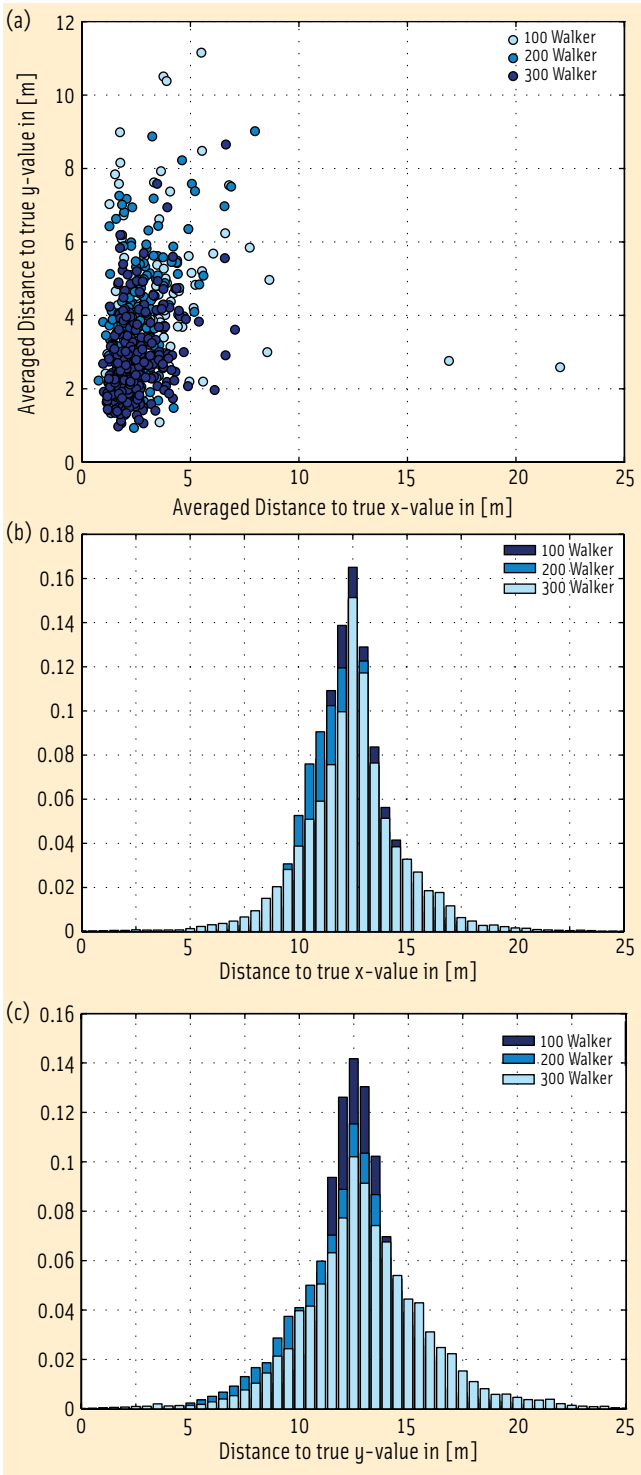

FIGURE 8 The distance of the estimated position to the true position for the simulation with an area of 100 x 200 meters and a correction from other walkers: (a) shows the averaged absolute distance to the true position for each walker for the whole simulation; (b) and (c) display the frequency distribution of the distance to true x- and y-values.

only correct at reference positions, especially when few other walkers are in the area.

However, as soon as the size of the weak-signal area increases and the amount of reference signals stays the same

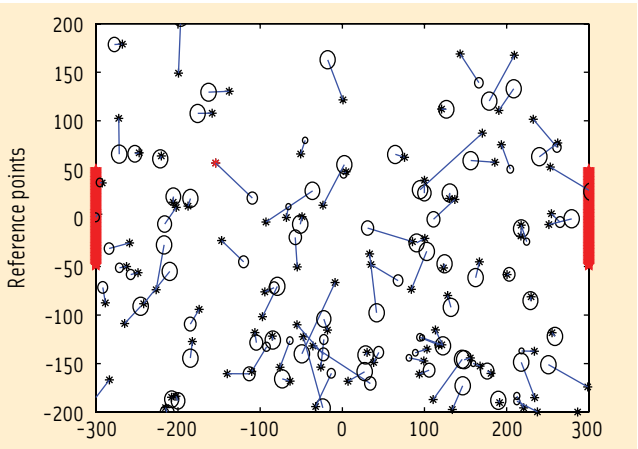

FIGURE 9 The indoor area for the second part of the simulation. The area has dimensions of 400 x 600 meters. The number of reference points remains the same.

or becomes less, the correction with other walkers gets more and more valuable.

The quantity of walkers moving around in the area also affects the outcome. Of course, the correction is better when a walker can choose the best (in this case, the nearest) of surrounding walkers.

Finally, it is obvious that this approach also has some drawbacks. These and the practicability of such a service in real life applications are discussed next.

Conclusions

The results of the simulations presented in this article indicate that our approach has its advantages in case a GNSS signal is available only intermittently or unavailable for an extended period of time. This peer-to-peer model works best in areas of great size and when the possibility of using GNSS is limited to a few spots. But it is important that enough users participate.

Our first simulations confirm that, with fewer than 100 walkers, no real improvement occurs compared to the situation when the correction is only done at the reference points. In practical terms, this means that the more people there are using this kind of navigation aiding, the better the service will work.

The use of Bluetooth for this kind of approach also seems logical because no additional costs are incurred when transmitting data. The master/slave architecture makes it easy for one device to start a piconet and to query different slaves to find the one that fits best to correct the position.

This peer-to-peer approach requires no additional infrastructure. The piconets can be easily and quickly implemented and disappear just as quickly. And, as already mentioned, Bluetooth is a very common protocol with widespread availability on portable devices.

It does not matter what kind of devices communicate with each other, making such a service potentially available for every portable device. With SDP (Service Discovery Protocol) Bluetooth has also already implemented a protocol that makes it easy to discover new services on other devices.

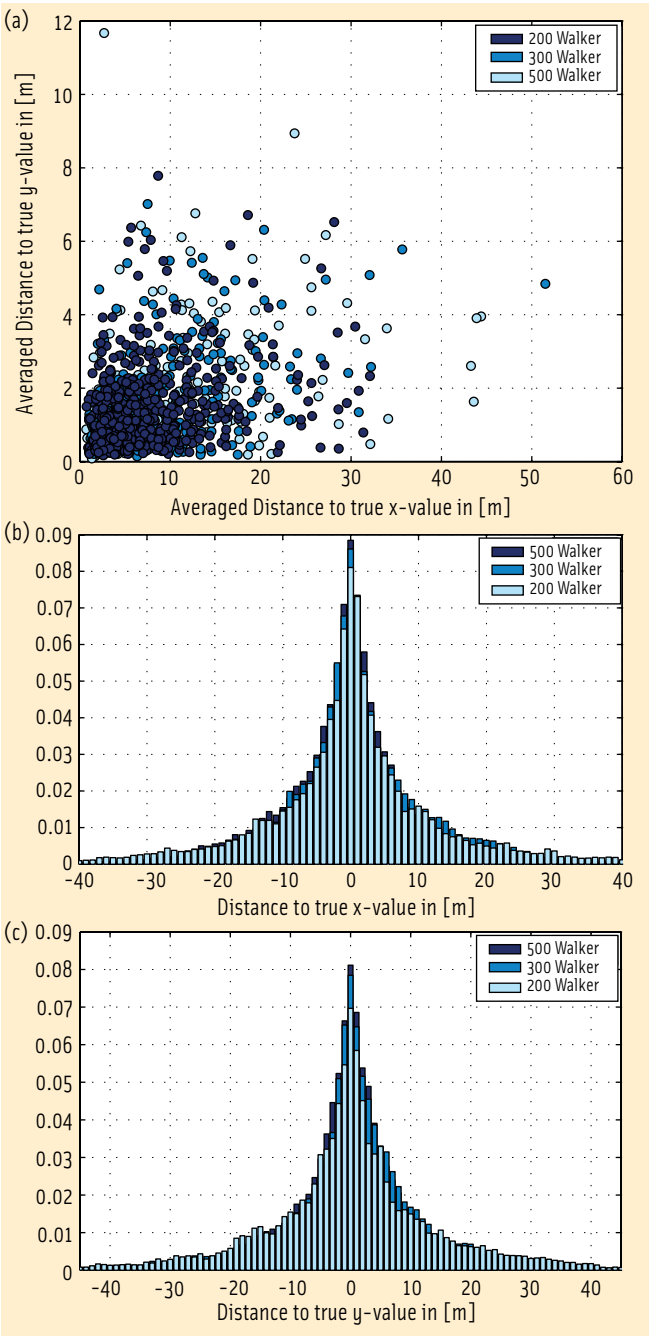

FIGURE 10 The distance of the estimated position to the true position for the simulation with an area of 400 x 600 meters and the correction with other walkers: (a) shows the averaged absolute distance to the true position for each walker for the whole simulation; (b) and (c) display the frequency distribution of the distance to true x- and y-values.

Nonetheless, some practical issues do arise. GNSS positioning is anonymous, but this kind of A-GNSS approach by definition requires the sharing of position-related information with neighboring users. A Bluetooth device must be enabled to participate in this service at any time, and the estimated positions of two walkers must be exchanged to calculate a correction. This means that incoming connections have to be accepted no matter who requests them.

Bluetooth, however, has several security mechanisms available to protect the user from wiretapping of their transmitted data: encryption and a challenge-response algorithm to authenticate the devices mutually. Normally each user has to enter a PIN to establish a connection with another device when the devices do not know each other. Based on this PIN, a link key is calculated and used for the rest of the connection session.

This security mechanism is often eroded by the manufacturers, which may allow a PIN of only four figures or implement a static PIN. But even when users are able to enter the PIN manually and begin the authentication process, they have to trust the other user's device when exchanging the PIN.

GNSS positioning is anonymous, but this kind of A-GNSS approach by definition requires the sharing of position-related information with neighboring users.

A better situation would be to have this process take place automatically when the user is not even aware of the data transmission between the devices to correct a position. Of course, the user has to authorize its device to take part in this kind of service. But the transmitted positions of other devices should not be known to him and should not be accessible in any of his mobile device's other applications. In this way, the other devices remain anonymous to the user.

Some variations on our approach come to mind. As mentioned earlier, the users' only additional device is a compass to measure their heading with a known accuracy and a step detector. The stride length is always thought to be 0.7 meters but surely varies depending on the walker. Adding a device that measures the stride length of each walker to a certain degree of precision would not be problematic. This would also make the position estimation more accurate when correcting only with reference positions.

Our original simulations had the walker correcting its position with the first walker that came within communication range. This often did not lead to the expected improvement in positioning accuracy. Indeed, this is the reason a walker now corrects its position with the one that is nearest. But we could also choose the walker with the smallest error variance or the walker with the minimum error. More simulations will be carried out to evaluate the decision rule that fits this approach best.

It might also be difficult to define which of the other walkers is the nearest. The only value that we can rely on for this is the *RSSI*, which as mentioned previously does not allow very precise conclusions about the distance between sender and receiver. An exchange of the error variance would provide an easier and more reliable approach.

In contrast to other approaches that use dead reckoning, Kalman filtering, or Bluetooth triangulation to acquire a position, this peer-to-peer approach needs neither many additional

devices nor a previous surveying of the area. The algorithm is not too complex and doesn't need much processing power — an important consideration particularly for smaller devices. By using Bluetooth as a transmission protocol the exchange of data is done in a local network without requiring or overloading an infrastructure network.

Applying Bluetooth operating in the 2.4 GHz portion of the S-band as a communication link to enhance navigation is also interesting from another point of view. As indicated in the article by J.A. Avila-Rodriguez et alia (Additional Resources), S-band could be used on a global basis for RDSS applications if approved by the International Telecommunications Union.

Indeed, both the Indian Regional Navigation Satellite System and China's Compass/Beidou GNSS are already using or will eventually use S-Band for navigation purposes on a regional basis. Using Bluetooth communications to support and enhance navigation performance in indoor environments could provide a more seamless link between a potential S-band GNSS navigation service and the communication link. This might encourage users to implement such a service on their mobile device.

Acknowledgments

I want to thank Professor Eisfeller and my colleagues at the Institute of Geodesy and Navigation, especially José-Angel Ávila-Rodríguez, Felix Kneissl, Herbert Niedermeier, and Stefan Wallner. A special thank you goes to my former colleague Thomas Pany who — at least at the beginning — knew better than me how to implement such a service.

References

[1] F. van Diggelen: *A-GPS Assisted GPS, GNSS, and SBAS,* Artech House, 2009

[2] IEEE Std 802.15.1: *Part15.1: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs),* 2002

[3] J. Schiller, *Mobilkommunikation,* Addison-Wesley, 2003, 2nd edition

[4] S. Feldmann, K. Kyamakya, A. Zapater, Z. Lue: *An indoor Bluetooth-based positioning system: concept, Implementation and experimental evaluation,* Technical Report, University of Hannover, 2003

[5] K.Thapa, S.Case: *An Indoor Positioning Service for Bluetooth Ad Hoc Networks,* MICS 2003, Duluthm, MN, USA

[6] E. Pulido, R. Quiros, H. Kaufmann: *Analysis of a Kalman Approach for a Pedestrian Positioning System in Indoor Environments*, In Proceedings of the European Conference on Parallel and Distributed Computing, 2007

[7] G. Lachapelle: *GNSS Indoor Location Technologies,* Journal of Global Positioning Systems Vol. 3, 2004

[8] P. Maybeck: *Stochastic Models, Estimation, and Control,* Vol.1 Chapter 1, Academic Press Inc, 1982

[9] G. Welch, G. Bishop: An Introduction to the Kalman Filter, SIGGRAPH Course, 2001

[10] Ch. K. Chui, G. Chen: *Kalman Filtering with Real-Time Applications*, Springer Verlag Berlin, 1987

[11] J.A. Avila-Rodriguez, S. Wallner, G.W. Hein, B. Eissfeller, M. Irsigler, J.L. Issler: *A Vision on New Frequencies, Signals and Concepts for Future GNSS Systems,* Proceedings of ION-GNSS 2007, September 2007, Fort Worth, TX, USA

[12] O. Mezentsev: Sensor Aiding of HSGPS Pedestrian Navigation, UCGE Reports, March 2005, Calgary Canada

[13] J. W. Kim, H. J. Jang, D. Hwang, Ch. Park: *A Step, Stride, and Heading Determination for the Pedestrian Navigation System,* Journal of Global Positioning Systems, 2004

[14] S. Lee, K. Mase, K. Kogure: *Detection of Spatio-Temporal Gait Parameters by Using Wearable Motion Sensors,* Proceedings of the 2005 IEEE, Engineering in Medicine and Biology 27th Annual Conference, September 2005, Shanghai, China

[15] H. Niedermeier, G. Ameres, Th. Pany, B. Eissfeller, J. Winkel, G. Lopez-Risueno: *Reproduction of User Motion and GNSS Signal Phase Signatures Using MEMS INS and a Pedestrian Navigation System for HS-GNSS Applications,* Proceedings of the ESA NAVITEC 2008 Conference, December, 2008, Noordwijk, Netherlands

Authors

Isabelle Krämer is a Research Associate of the Institute of Geodesy and Navigation at the University FAF Munich. She got her diploma in Computer Science from the Ludwig-Maximilian-University Munich. She works in the Software Receiver group. Her main research interests are A-GNSS and developing networks for several software GNSS receivers.

Bernd Eissfeller is Full Professor and Director of the Institute of Geodesy and Navigation at the University of the Federal Armed Forces in Munich. He is responsible for teaching and research in the field of geodesy, navigation and signal processing. He worked in industry as a project manager on the development of GPS/INS navigation systems until 1993. From 1994 to 2000 he was head of the GNSS Laboratory. From 2000 to 2008 he was Vice Director of the IGN until he became the Director of the Institute which was formerly led by Prof. Guenter Hein.

"Working Papers" explore the technical and scientific themes that underpin GNSS programs and applications. This regular column is coordinated by **PROF. DR.-ING. GÜNTER HEIN.** Prof. Hein is a member of the European Commission's Galileo Signal Task Force and organizer of the annual Munich Satellite Navigation Summit. He has been a full professor and director of the Institute of Geodesy and Navigation at the University of the Federal Armed Forces Munich (University FAF Munich) since 1983. In 2002, he received the United States Institute of Navigation Johannes Kepler Award for sustained and significant contributions to the development of satellite navigation. Hein received his Dipl.-Ing and Dr.-Ing. degrees in geodesy from the University of Darmstadt, Germany. Contact Prof. Hein at <Guenter.Hein@unibw-muenchen.de>. IG