Metrol. Meas. Syst., Vol. XVIII (2011), No. 1, pp. 145–158

METROLOGY AND MEASUREMENT SYSTEMS

Index 330930, ISSN 0860-8229
www.metrology.pg.gda.pl

FUSION OF DATA FROM INERTIAL SENSORS, RASTER MAPS AND GPS FOR ESTIMATION OF PEDESTRIAN GEOGRAPHIC LOCATION IN URBAN TERRAIN

Przemyslaw Baranski, Maciej Polanczyk, Pawel Strumillo

Technical University of Lodz, Institute of Electronics, 211/215 Wolczanska, 90-924 Lodz, Poland
(*przemyslaw.baranski@p.lodz.pl, maciej.polanczyk@p.lodz.pl, ✉pawel.strumillo@p.lodz.pl*, +48 42 631 26 46)

Abstract

An electronic system and an algorithm for estimating pedestrian geographic location in urban terrain is reported in the paper. Different sources of kinematic and positioning data are acquired (i.e.: accelerometer, gyroscope, GPS receiver, raster maps of terrain) and jointly processed by a Monte-Carlo simulation algorithm based on the particle filtering scheme. These data are processed and fused to estimate the most probable geographical location of the user. A prototype system was designed, built and tested with a view to aiding blind pedestrians. It was shown in the conducted field trials that the method yields superior results to sole GPS readouts. Moreover, the estimated location of the user can be effectively sustained when GPS fixes are not available (e.g. tunnels).

Keywords: pedestrian navigation, data fusion, GPS, digital maps, particle filtering, electronic travel aids.

1. Introduction

Reliable and accurate estimation of outdoor personal positioning is an important problem in many professional and leisure activities. Applications are numerous, e.g.: tracking of soldiers or firemen in action [1], aiding travel for the visually impaired [2], locating the whereabouts of children and the elderly [3], or navigating tourists around points of interest [4].

In open outdoor spaces, an obvious solution is to use the Global Positioning System (GPS) that, together with data derived from digital maps, offers a powerful positioning and navigation tool. Car navigation devices mostly implement such solutions. The available software is capable of improving positioning accuracy by using *a priori* information concerning the layout of road networks. However, a pedestrian navigation task poses a more complex challenge [5, 6, 7]. Firstly, the path the pedestrian walks is not confined to *a priori* given routes. Secondly, there are few maps available that provide detailed information about pavement layout in urban terrain (e.g. underground pathways, large squares, parks etc.). Thirdly, the mobility pattern of a pedestrian is more complex than that of the vehicles moving along road lanes. Hence, the routing algorithms used in car navigation procedures cannot be directly applied in pedestrian navigation. Finally, in the urban environments, pedestrians walk alongside buildings' walls making the GPS readouts even less accurate (the positioning errors can reach several dozens of meters). The indicated problems of pedestrian navigation tasks call for additional sources of positioning data that could be fused with GPS readouts and enhance the navigation performance of a pedestrian. There have been a number of reported systems aiding GPS based pedestrian navigation that can be grouped into two major types:

Article history: received on Nov. 16, 2010; received in revised form on Feb. 10, 2011; accepted on Feb. 18, 2011; available online on Mar. 21, 2011; DOI: 10.2478/v10178-011-0014-3.

- smart environment systems: in which a dedicated infrastructure is embedded into the urban terrain or alternatively an already existing telecommunication system (GSM base stations) are used for positioning,
- wearable sensors that are carried by the navigated user.

In the smart environment systems, a network of electronic tags (radio, ultrasound or infrared beacons) is embedded into the environment. Example solutions employ Radio Frequency Identification Devices (RFID) [8] or infrared based systems like the Talking-Signs [9] designed for navigating the visually impaired. The pedestrian position is tracked by the beacons whose landmark is known. Unfortunately, such systems are costly in implementation. Large number of tags need to be mounted and serviced, although the unit cost of such devices is consistently dropping. In smart environment systems, a pedestrian's position can be determined by multilateration algorithms, similar to the ones used in the GPS [10].

Wearable sensors are becoming an attractive mobile solution for aiding pedestrians. Currently, such devices as accelerometers, electronic gyroscopes and compasses, digital cameras and barometers are of small size and consume low power. Some of the devices can be integrated into cloths. The wearable sensors allow estimation of a pedestrian's kinematical parameters, his cardinal directions or local positioning against the surrounding objects. However, due to limited accuracy of the sensors and an iterative manner of motion estimation, special dead-reckoning algorithms need to be employed. Dead-reckoning can be defined as the task of determining own (e.g. pedestrian) position on the basis of a previous fix and current kinematical parameters (e.g. speed, movement angle).

Aided navigation can only be successfully implemented if various sources of positioning or kinematic data can be suitably fused and continuously updated for efficient position tracking of a moving object [10].

The paper proposes an iterative Monte-Carlo method (also known as particle filtering) that fuses and processes the data from an accelerometer, gyroscope, GPS receiver and raster maps. The accelerometer serves as a step meter for calculating the number of steps as well as estimating their length [11]. The gyroscope provides the angular velocity which is used to calculate a relative rotation of the traveller. Should the GPS fixes diverge, the user's position is predicted in a dead-reckoning manner from the step meter and gyroscope readouts. The raster map of the terrain is converted into a probability map, in which different probability values are assigned to the different regions the user passes (e.g. who is more likely to walk along a pavement than traverse walls, fences or ponds). In a number of field trials, it was shown that the built prototype system and the proposed data fusion algorithm significantly improved the estimation of a pedestrian's geographic location in urban terrain.

2. System components

The system block diagram is shown in Fig. 1. A small portable laptop serves the purpose of a computational unit. The system can estimate location on-line as well as store data from the sensors for off-line analysis.

All the sensors (accelerometer, gyroscope and GPS) are encompassed in a small PCB board ('Sensor board' see Fig. 1) that connects to the PCB via an USB interface. The sensors board can work independently storing data on a Micro-SD card. The board can be then connected to the PC and all recorded data can be read off. Fig. 2 shows a picture of the sensors PCB.

The PC records on-line the data stream from the sensors board and processes them. The accelerometer readouts are digital-filtered to estimate the number of steps and their length. The angular velocity, provided by the gyroscope, is integrated and a relative rotation angle is calculated. The step lengths and rotations thus determine a relative user displacement, which is used to predict the geographical location should the GPS fixes be inaccurate.

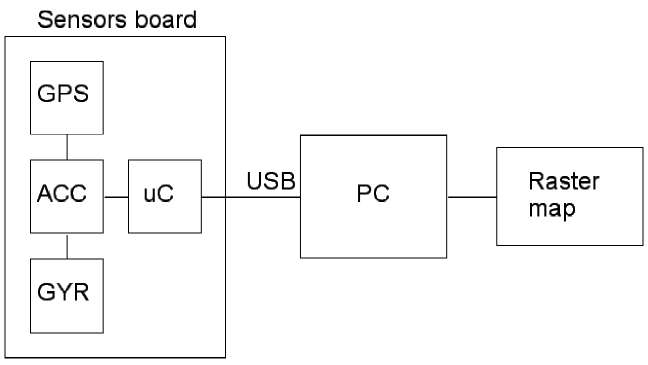

Fig. 1. System block diagram, ACC and GYR stands for accelerometer and gyroscope respectively and uC denotes a microcontroller.

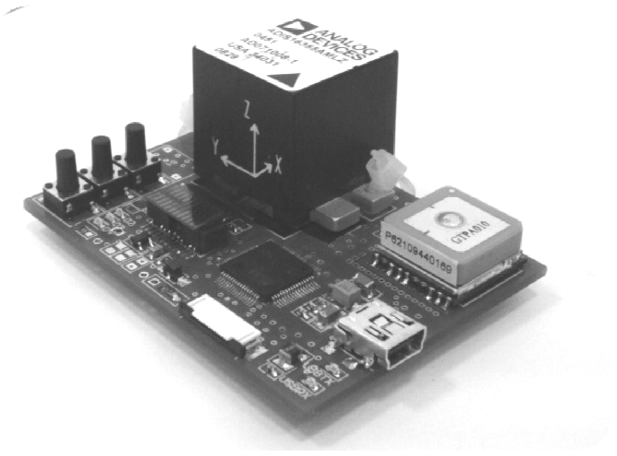

Fig. 2. The sensors board (65mm×48mm×29mm). The large cube is a digital 6DOF sensor (Analog Devices ADIS16355) that houses 3-axis accelerometer and 3-axis gyroscope. The smaller cube is a GPS receiver (with an integrated antenna). The bottom layer comprises a Micro-SD slot and a battery connector for the independent-work mode. When connected to a PC, the power is drawn from the USB interface.

2.1. The GPS receiver

The FGPMMOPA6B GPS receiver comes with an integrated ceramic antenna of compact measurements (16mm x 16mm). Apart from the location estimates, every GPS receiver within the NMEA protocol provides a parameter that reflects the quality of a fix – i.e. HDOP (Horizontal Dilution of Precision) [10]. Larger values of this parameter correspond to less accurate estimates. This parameter is crucial with a view of the whole algorithm that is to take decision whether to believe the GPS receiver or predict the location from the step meter, gyroscope and raster maps. We categorized the value of HDOP parameter into three ranges:

- HDOP < 1.0 – accuracy better than 5m
- HDOP < 2.0 – mediocre accuracy, better than 15m
- HDOP ≥ 2.0 – poor accuracy, worse than 15m.

These values differ greatly from receiver to receiver. We tested three GPS receivers: HAiCOM HI-204III, Pentagram Pathfinder P3105 and FGPMMOPA6B which returned competently different values in the same locations at the same time.

It was noticed that the HDOP parameter, in many cases, does not reflect the actual precision, which initially complicated the implementation of the algorithm. To account for such events, after a period of poor quality estimates, we introduce the so called guard time, i.e. the value of HDOP is filtered so that it does not abruptly change from high to low values. The following simple rule was applied for HDOP filtering:

IF (current_HDOP < previous_HDOP) THEN current_HDOP :=
previous_HDOP + A˙(previous_HDOP – current_HDOP)

Factor *A* ranges from 0.05 to 0.1 and depends on the GPS receiver model. Lower values of this parameter correspond to a longer guard period. In Fig. 3 an example of filtering the HDOP value by using the proposed rule is shown.

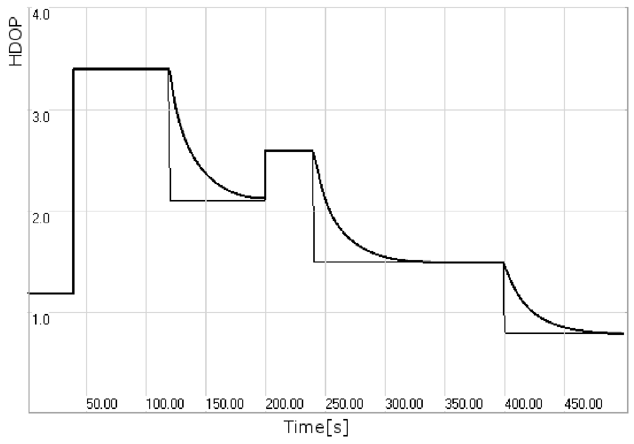

Fig. 3. Filtering the HDOP parameter: thin line corresponds to HDOP values obtained from the GPS receiver and thick line denotes the filtering result.

The implementation of this non-linear filtering method has improved overall system performance. However, unique situations take place in which inaccurate values of HDOP confuse the system.

2.2. The accelerometer

The data from the accelerometer is used to estimate the length and number of steps of a person. The sample rate is 50Hz. The resolution of the readouts is 14-bits per sample. As a person walks the body moves up and down in accord with the steps. The step detection algorithm examines the acceleration of a body in the vertical direction only (i.e. parallel to the gravity axis). The algorithm estimates step lengths with 1.5% accuracy when the user walks on a flat ground. The step detection algorithm is shown in Fig. 4. A detailed explanation of the algorithm together with the tests performed on a number of pedestrians follow in the authors' paper [11].

The pedometer is sensitive to orientation, as it measures the acceleration in a given axis to estimate the step length. Any deviation $ø_d$ from the direction of the gravity will scale the accelereometer's readouts by $cos(ø_d)$, as it is clarified by Fig. 5. According to the step estimation algorithm, the lengths of steps will be scaled by $cos^{0.25}(ø_d)$. The estimated step length is then given by formula:

$$d = K \cdot (a_{m\max} - a_{m\min})^{0.25} = K \cdot (a_{g\max} - a_{g\min})^{0.25} \cos^{0.25}(\varphi_d)\,, \tag{1}$$

where: $a_{m\max}$ and $a_{m\min}$ denote the measured peaks in the acceleration (cf. Fig. 4), $a_{g\max}$ and $a_{g\min}$ stand for the peaks of acceleration in the gravity axis.

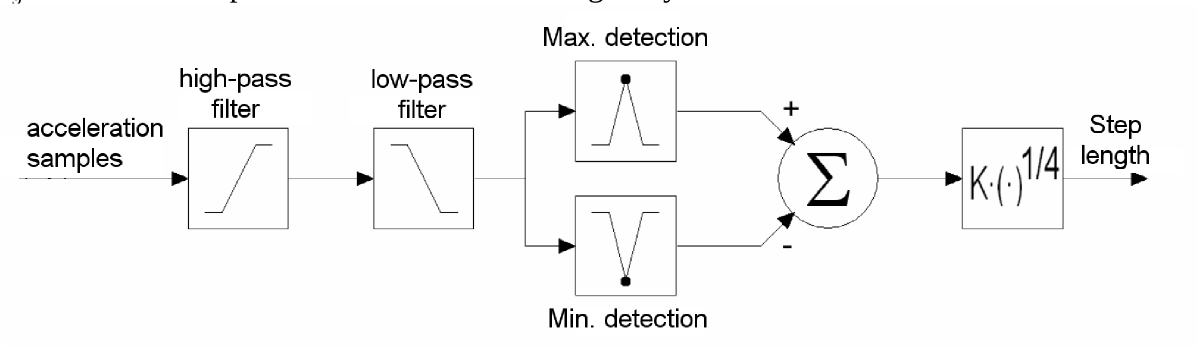

Fig. 4. Step detection algorithm. The acceleration samples are filtered to reject undesired constituents like body swings and so on. A maximum and minimum of accelerations are detected and multiplied by a scaling factor *K*, which is individual for a given person. Then the result is raised to an empirical power of 0.25.

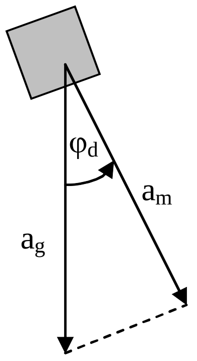

Fig. 5. A sensor misalignment causes that the sensors measures the acceleration a_m which is equal $a_g \cdot \cos(\varphi_d)$.

The error due to the sensor misalignment is given by the following formula:

$$\frac{\partial d}{\partial \varphi_d} = \frac{K \cdot (a_{g\max} - a_{g\min})^{0.25}}{\cos^{0.75}(\varphi_d)} \sin(\varphi_d)\ . \tag{2}$$

For small reasonable angles $ø_d$, the algorithm accuracy is not significantly compromised. For example, for $ø_d$=10^o, the step length is scaled by 0.996.

It was noticed during tests that tripping over had an influence on the algorithm performance. These events caused the sensor to register 1 to 4 spurious steps. Moreover, blind testers were noticed to make steps in a place when they were not sure were to go. The latter error can be eliminated by measuring the acceleration in the walking direction. This solution has yet to be implemented. The accelerometer readouts are perturbed by white noise [12]. As the distance estimation is based on the previous one and the current step length, the whole process is described by a random walk, whose variance grows unboundedly over time. This will be explained in the next chapter which covers the gyroscope accuracy. The accelerometer bias instability does not play an important role, because the step lengths are calculated as the difference between subsequent extrema. Calculating the displacement of the user by integrating twice the acceleration will quickly lead to enormous errors. Any bias that may appear due to e.g. sensor misalignment or temperature change, results in the displacement error growing quadratically with time.

2.3. The gyroscope

The data from the gyroscope is integrated to calculate a relative rotation angle $\Delta\phi$, according to the following formula:

$$\Delta\phi(t) = \int_0^t \left(\omega + \omega_{drift}\right) \cdot dt\,, \tag{3}$$

where: t – is the integration time, i.e. the sampling interval, ω – is the angular velocity of a body, ω_{drift} – is the angular drift velocity.

Every gyroscope exhibits a so called drift, i.e. the sensor shows an angular velocity despite no movement. This depends on the temperature, sensor alignment and circuit voltage supply. The manufacturer claims that the gyroscope is temperature compensated. Nonetheless the drift was approx. 0.2°/s. Thus in the time period of 15minutes the gyroscope can stray 180° from the true direction. The systematic bias can be easily measured and subtracted. Similarly to the accelerometer, the gyroscope measurements $\omega(t)$ are flawed by white noise $n_\omega(t)$. The estimated angle $\Delta\varphi(t)$ is then corrupted by the error expressed by the formula:

$$\int_0^t \left(n_\omega(t)\right) \cdot dt \approx \sum_{i=1}^{n} N_\omega(i)\delta t\,, \tag{4}$$

where: δt is the sampling period, $N_\omega(i)$ are samples of angular speed $\omega(t)$, i is the measurement number, whereby $t = n \cdot \delta t$. The variance of the aforementioned error (of zero mean) can be calculated according to the following formula:

$$Var\left\{\sum_{i=1}^{n} N_\omega(i)\delta t\right\} = E\left\{\sum_{i=1}^{n}\left(N_\omega(i)\delta t\right)^2\right\} = n \cdot \delta t^2 \cdot E\left\{N_\omega(i)^2\right\} = n \cdot \delta t^2 \sigma_\omega^2 = t \cdot \delta t \cdot \sigma_\omega^2\,, \tag{5}$$

where: $\sigma_\omega^{\ 2}$ is the variance of the angular velocity measurements. The variance of the estimated angle $\sigma_\varphi^{\ 2}$ grows boundlessly with the time. The manufacturer of the sensor [12] states that the deviation σ_φ equals $4.2^0/\sqrt{h}$, where h – is given in hour time units. Hence, after one hour the deviation equals σ_φ=4.2° and after two hours reaches σ_φ=5.8° etc. σ_φ grows proportionally with time due to bias instability at the level of 54°/*h* [12]. There are other factors, however, of much lesser importance, that degrade the gyroscope's performance like: change of gyroscope sensitivity due to temperature, axis nonorthogonality, nonlinearity, voltage sensitivity etc. What should be also taken into account is the sensor misalignment which causes the readouts to be scaled, similarly as with the accelerometer. Laser gyroscopes improve by far on MEMS gyroscopes. For example, GG1320AN Digital Laser Gyro is a single axis gyroscope with a bias instability of 0.0035°/*h* and an angular random walk of $0.0035^0/\sqrt{h}$. The dimensions rather disqualify them from mobile applications – 8*cm*×8*cm*×4.5*cm*. By comparison, the ADIS16355 measures 2*cm*×2*cm*×2*cm*. The cost of the laser gyroscope is at least several times higher than the MEMS (Micro Electro-Mechanical Systems) based device ADIS16355.

The authors performed also tests with an electronic compass which provides absolute orientation. The compass was very susceptible to any magnetic field distortions caused by cars, power lines, trams etc. For example, a tram, 30 meters away, rendered the compass readouts useless for around 30s [11].

Metrol. Meas. Syst., Vol. XVIII (2011), No. 1, pp. 145–158

2.4. The raster map

The raster map is divided into three distinct regions with different weights assigned as follows:

– forbidden areas – buildings, ponds, walls, fences etc., w_{map}=0.1,
– probable areas – lawns, fields etc., w_{map}=0.5,
– preferred areas – pavements, streets, squares, alleys etc., w_{map}=1.0.

Each weight corresponds to the probability of the user being in a given region. The probability map mitigates the problem of gyroscope drifts and step meter inaccuracies. The weight value for the forbidden regions (w_{map}=0.1) was chosen to provide some “flexibility” for the data fusion algorithm. Initially we set w=0 for the forbidden regions. However, cases occurred, due to inadequate HDOP parameter readouts, that the estimated locations (particles) “enter” the wall and die out due to the zero weight value assigned. Then all the particles had to be randomly reinitialized. Hence, the algorithm required extra time to converge to an optimal solution, in which case it worsened the overall computational performance. The values of the map’s weights were assumed empirically on the basis of different trials. The environment is subject to change, e.g. a new pavement, road and a digital map is not always up to date. The weights should provide some margin for that. A small value for probable areas would cause that the particles would die out quickly e.g. when a user crosses a wide street and wants to travel on the other side. The algorithm would then estimate that the pedestrian is on the wrong side. Fig. 6 shows an example of a raster probability map.

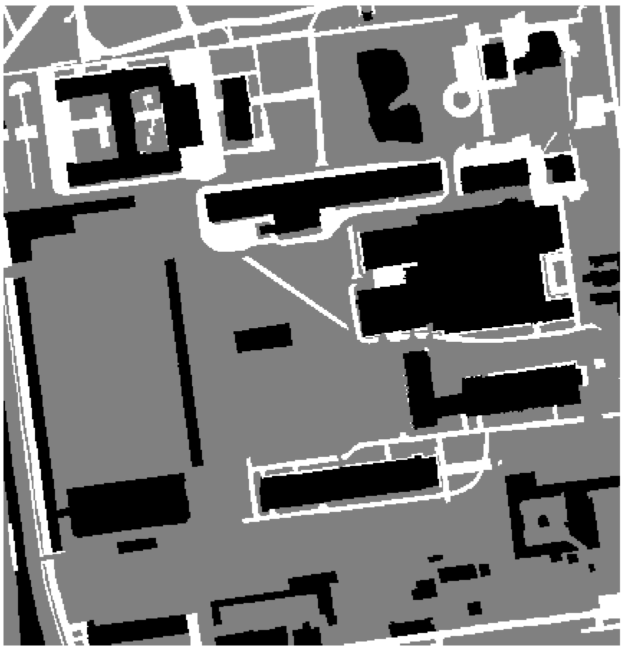

Fig. 6. A raster map of the University campus. White, grey, black correspond to preferred, probable and forbidden regions respectively.

3. The particle filtering algorithm applied to positioning estimation

3.1. Introduction

Particle filtering is an advanced model estimation technique derived from a sequential version of Monte Carlo methods [13, 14]. This approach uses a simulation technique that generates a large number of candidate solutions (particles) in search for the best solution to a complex problem. An estimated system state is represented by a large number of particles. Each particle undergoes certain rules and then it is assessed how accurately it approximates the solution (measurement update). Therefore, every particle has an associated weight (importance). The final system state is a weighted average of particle states (prediction). With successive iterations of the simulation, some particles assume negligible weights and do not effectively take part in the simulation. These particles are replaced with the ones that more accurately converge to the optimal solution – this is the so-called resampling technique [13].

Kalman filters are commonly used for dead reckoning applications. The Kalman filter is a recursive algorithm, whereby the current state depends only on the previous one and the driving inputs [15]. The filter is much less computation expensive than a particle filter. When the object model is nonlinear, then the Extended Kalman Filter (EKF) is applied. The filter is based on linearization around the current state by the Taylor approximation. The algorithm performs well when the system exhibits moderate nonlinearity and moderate non-Gaussian measurement noise. Due to linearization, the covariance matrix is propagated with an error which grows with every iteration. The algorithm, called the Unscented Kalman Filter (UKF), deals with significant non-linearities by sampling the probability distribution around the mean in so-called sigma points [15]. The above discussed techniques can effectively be used when the posterior distribution is unimodal [16]. Often, there are situations when two solutions are possible, for example, a pavement can bifurcate into two paths of a similar orientation. The traveller might be on either path but not between them, as a Kalman algorithm would estimate. One path can be pointed as more probable but at a later time. For these kinds of problems, the ensemble Kalman filter, also referred to EnKF, is employed [17]. The system state is represented by a large collection of ensemble members. Each member implements the idea of the Kalman filter and it is individualized by adding a small perturbation to its state. The EnKF works under the assumption that all probability distributions are Gaussian [18]. As a matter of fact, the EnKF is closely related to the particle filter, which can assume any probability distribution, albeit at a heavy computational cost. The main advantage of a particle filter is quick implementation and flexibility when adding new data sources.

3.2. *Implementation of the particle filtering algorithm*

In our simulation, every particle c_i (i – is particle number) that takes part in the simulation is a hypothetic user location, thus it can be described by the following vector denoted by formula 6.

$$c_i = [x_i(t), y_i(t), \varphi_i(t), w_i(t)], \tag{6}$$

where: $x_i(t)$, $y_i(t)$ – a particle location in Cartesian coordinates, $\varphi_i(t)$ – an angle associated with the particle, i.e. angle versus the coordinates, $w_i(t)$ – a particle weight. In the discussed application the number of particles must be at least 2500 to obtain good simulation accuracies.

For the algorithm outset, all the particles are initialized with randomly distributed angles $\varphi_i(t)$, and the coordinates $x_i(t)$, $y_i(t)$ are spread around GPS estimates converted into Cartesian coordinates $x_{GPS}(t)$, $y_{GPS}(t)$. This algorithm step is called *initialization.*

As the user walks, the step meter and gyroscope provide data about relative user displacement – $d(t)$ and $\Delta\varphi(t)$ respectively. Every particle is updated with this data. Moreover, the algorithm simulates sensor inaccuracies by introducing measurement errors. The algorithm generates (randomly) measurements errors individually for every particle. A Gaussian noise model was used both for the step meter and the gyroscope. A particle transition from the current to the next state is explained in Fig. 7.

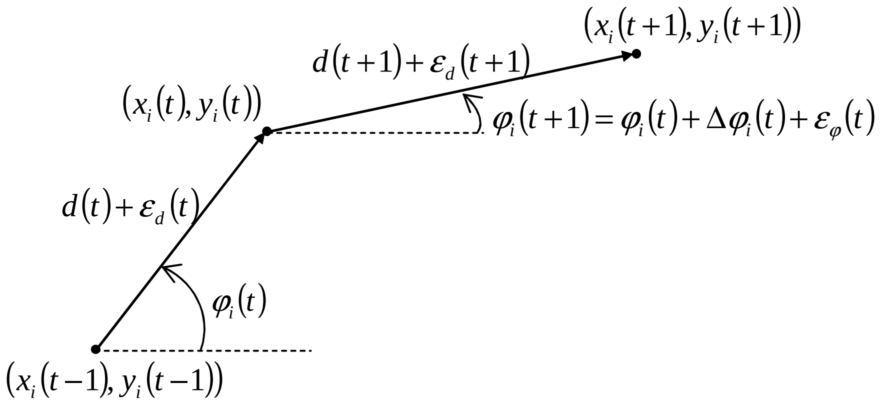

Fig. 7. Particle transition introduced by the step meter and gyroscope. $d(t)$ is a step length, $\Delta\varphi(t)$ is the relative rotation. $\varepsilon_d(t)$ and $\varepsilon_\varphi(t)$ are simulated measurement errors for the step meter and gyroscope respectively. The algorithm generates these values randomly for every updated particle.

Following Fig. 7, a particle transition to a new state is given by equation (7).

$$\begin{bmatrix} x_i(t+1) \\ y_i(t+1) \end{bmatrix} = \begin{bmatrix} x_i(t) \\ y_i(t) \end{bmatrix} + (d(t)+\varepsilon_d(t)) \begin{bmatrix} \cos(\varphi_i(t)+\Delta\varphi(t)+\varepsilon_\varphi(t)) \\ \sin(\varphi_i(t)+\Delta\varphi(t)+\varepsilon_\varphi(t)) \end{bmatrix}. \tag{7}$$

Then every particle is assessed, i.e. how accurate it approximates the user location. The assessment is based on GPS measurements and the map. If a particle encounters a forbidden region and in the long run does not coincide with GPS measurements, it means that the particle has a wrong orientation $\varphi_i(t)$ or its location $(x_i(t),y_i(t))$ is erroneous. The weight $n_i(t)$ of this particle is accordingly lowered.

The weights of particles are updated on the basis of GPS measurements according to equation (8).

$$w_i(t+1) = w_i(t) \cdot PDF_{GPS}(\delta_i(t+1)), \tag{8}$$

where $PDF_{GPS}(\delta_i(t))$ is a probability density function for the $e_i(t)$ error. We used a normal error distribution, expressed by equation (9).

$$PDF_{GPS}(\delta_i(t)) = \frac{1}{\sigma_{GPS}\sqrt{2\pi}} \exp\left(-\frac{e_i(t)^2}{2\sigma_{GPS}^{\ 2}}\right). \tag{9}$$

σ_{GPS} is standard deviation of GPS measurements. Empirically we assumed that σ_{GPS} is proportional to the squared HDOP parameter. $e_i(t)$ is the distance between a given particle location $(x_i(t),y_i(t))$ and the GPS measurement $(x_{GPS}(t),y_{GPS}(t))$, thus it can be calculated from equation (10).

$$e_i(t) = \sqrt{(x_i(t) - x_{GPS}(t))^2 + (y_i(t) - y_{GPS}(t))^2}\ . \tag{10}$$

The raster map is used also to update particle weights – equation (11).

$$w_i(t+1) = w_i(t) \cdot w_{map}(x_i(t), y_i(t)), \tag{11}$$

$w_{map}(x_i(t),y_i(t))$ is an area weight for given coordinates $(x_i(t),y_i(t))$ (see section *2.4 The raster map*).

The weights of the particles are normalized according to formula 12.

$$w_i(t) := \frac{w_i(t)}{\sum_{i=1}^{L} w_i(t)}, \tag{12}$$

where L is the total number of particles.

After a number of algorithm iterations, all but a few particles have negligible weights and therefore do not participate in the simulation effectively. This situation is detected by calculating the so called degeneration indicator given by equation (13).

$$g(t) = \frac{1}{L\sum_{i=1}^{L} w_i(t)^2}. \tag{13}$$

If $g(t)$ falls below a given threshold of e.g. 0.6, then a process called *resampling* is incorporated into the algorithm in which a new set of particles is created. The probability of copying a particle to the new set is proportional to its weight $w_i(t)$. Therefore particles that poorly approximate the system state are replaced by more “accurate” particles. Then all particles are assigned the same weight L^{-1}.

The estimated system state, i.e. the user location $(x_u(t),y_u(t))$, is a weighted average of all particle states:

$$\left(x_u(t), y_u(t)\right) = \left(\sum_{i=1}^{L} x_i(t)\cdot w_i(t), \sum_{i=1}^{L} y_i(t)\cdot w_i(t)\right). \tag{14}$$

4. Results

Fig. 8 shows an example path followed by a pedestrian in urban terrain. The route was ca. 1.9km long. The histogram of geographic estimation errors for the path is shown in Fig. 9.

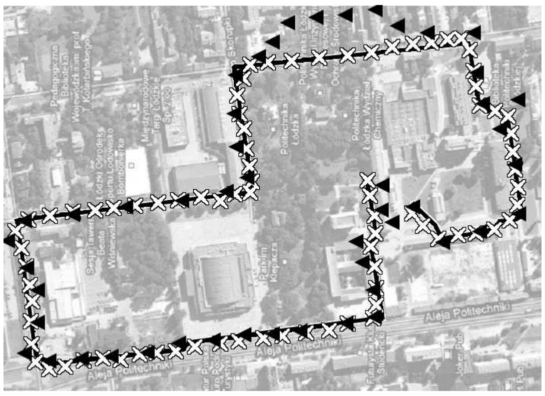

Fig. 8. A path followed by a pedestrian: Solid line –true path, crosses – estimated path, triangles – GPS readouts.

First, the absolute errors between the true location (read from a map) and the location estimated by means of the applied algorithms are calculated. Next, the errors are sorted with respect to their values that were expressed in meter units. The number of errors falling into each histogram bin is divided by the number of measurements and multiplied by a factor of 100. From the histogram shown in Fig. 9, we can conclude that the estimation algorithm (3) is the most precise, since more that 70% of location estimations are within the range of 0÷3m. In other words, the proposed algorithm brings large errors to smaller range. Hence, the number of smaller errors increases. It would be ideal if 100% of the errors fell in the range of 0÷3m.

Metrol. Meas. Syst., Vol. XVIII (2011), No. 1, pp. 145–158

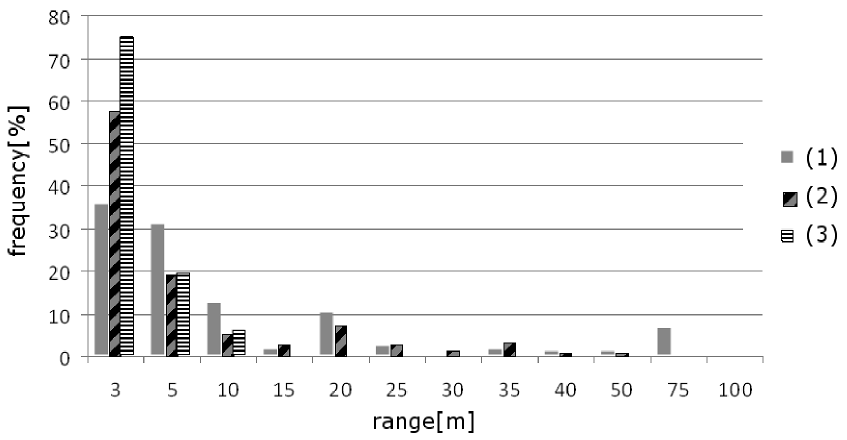

Fig. 9. The error histogram for the path. (1) – sole GPS readouts, (2) – GPS + inertial sensors + particle filtering, (3) – GPS + inertial sensors + map + particle filtering.

The histogram clearly shows that the proposed method combining data from inertial sensors and raster maps improves on GPS readouts. The errors were cut down to the range of up to 10 meters. This is a significant improvement, but it is still not enough to build a device that would robustly guide the user on a certain side of the pavement, in which case the accuracy must be within 2 meters. As shown in Fig. 9, implementation of inertial sensors improves on sole GPS readouts, by eliminating large errors (ca. 75m – cf. Fig. 10). Since the gyroscope provides only direction change and a new location is based on the previously estimated one, the predication strays away over time, as the errors quickly accumulate, e.g. when the direction is estimated with 10^{o} accuracy, after traversing a 100m distance the predicted location is ca. 20m from the correct location. Moreover, GPS errors in many cases tend to be systematically biased. Introduction of new data modality taken from the raster maps gives a synergetic benefit. As the particles attempt to drift away from the correct direction, they are trimmed by the building regions – cf. Fig. 10. In consequence, the direction is self-aligned and the prediction precision improves.

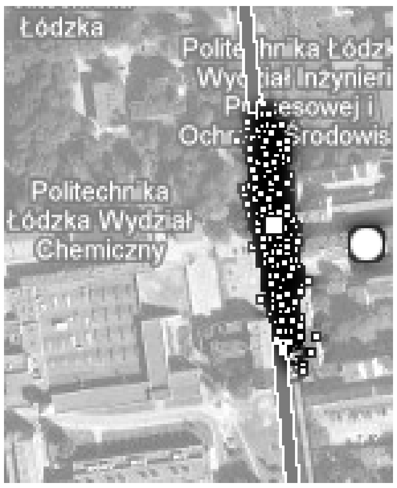

Fig. 10. A screenshot of the particle filtering simulation: large round dot – GPS location readout, large square – estimated user location, small squares – particles. Note that the algorithm eliminates the particles that would enter forbidden areas (i.e. buildings, fences, etc.).

5. Conclusions and future work

Application of inertial sensors and raster maps gives a measurable improvement in location estimation in urban terrain (Fig. 9). The errors were reduced several times – from ca. 70m to 10m. The presented example path (Fig. 8) was taken from one of the best results that were obtained. On several occasions, however, the particles entered an adjacent street or a square and encountered a dead-end in a form of connected buildings or walls. In these cases the algorithm was inferior to the GPS receiver. A reliable recovery algorithm needs to be worked out. It was noticed during the tests that the algorithm lacks parallelism. When the particles met several similar direction paths, they accordingly branched, but over time one path – not always the correct one – monopolized the particles. This led in many cases to the aforementioned deadlock. A larger number of particles alleviated the problem, but the computation burden was prohibitive for a real-time simulation. An additional source of data is currently considered to further improve positioning precision. Namely, the stereovision technique is being tested and compared against 3D maps of the terrain. The first successful field trials were conducted on the University Campus.

Acknowledgements

This work has been supported by the National Centre for Research and Development of Poland grant no. N R02 0083–10 in years 2010–2013.

References

[1] Fischer, C., Gellersen, H. (Jan.–Mar. 2010). Location and Navigation Support for Emergency Responders: A Survey. *Pervasive Computing*, 38–47.

[2] Hersh, M.A., Johnson, M.A. (2008). *Assistive Technology for Visually Impaired and Blind People*. Springer-Verlag. London Limited

[3] Helal, S., Mokhtari, M., Abdulrazak, B. (Eds). (2008). The Engineering Handbook of Smart Technology for Aging. Disability and Independence. John Wiley & Sons.

[4] Arikawa, M., Konomi, S. (Jul.-Sept. 2007). Navitime: supporting pedestrian navigation in the real world. *Pervasive Computing,* 21–29.

[5] Retscher, G., Thienelt, M. (2004). NAVIO – a navigation and guidance service for pedestrians. *Journal of Global Positioning Systems,* 3(1–2), 208–217.

[6] Mezentsev, O., Lachapelle, G., Collin J. (2005). Pedestrian dead reckoning – a solution to navigation in GPS signal degraded areas. *Geomatica,* 59(2), 175–182.

[7] Fang, L., Antsaklis, P.J., Montestruque, L., McMickell, M.B., Lemmon, M., Sun, Y., Fang, H., Koutroulis, I., Haenggi, M., Xie, M., Xie, X. (2005). Design of a Wireless Assisted Pedestrian Dead Reckoning System – The NavMote Experience. *IEEE Trans. on Instrumentation and Measurement,* 54(6), 2342-2358.

[8] Kourogi, M., Sakata, N., Okuma, T., Kurata, T. (2006). Indoor/Outdoor Pedestrian Navigation with an Embedded GPS/RFID/Self-contained Sensor System. *Proc. Int. Conf. Artificial Reality and Telexistence* CAT2006,1310-1321.

[9] The Smith-Kettlewell Eye Research Institute. (2010 November) *www.ski.org*.

[10] Farell, J. A. (2008). *Aided Navigation.* McGraw Hill. New York.

[11] Baranski, P., Strumillo, P. (2010). An electronic module implementing dead-reckoning algorithm for improving geographic location accuracy from GPS readouts in urban terrain. *VIII-th National Electronics Conference.* 07-10 June 2010. (in Polish).

[12] The website of Analog Devices. *www.analog.com/static/imported-files/data_sheets/ADIS16355.pdf*. (2011 February).

[13] Arulampalam, M.S., Maskell, S., Gordon, N., and Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans. Signal Processing*, 50, 174–188.

[14] Ceranka, S., Niedźwiecki, M. (2003). Application of Particle Filtering in Navigation System for the Blind. *Proceedings of the IEEE Seventh International Symposium on Signal Processing and its Applications.*

[15] Brown R. (1983). Introduction to Random Signal Analysis And Kalman Filtering. *John Wiley & Sohns*.

[16] Cappe O., Godsill S., and Moulines E. (2007). An overview of existing methods and recent advances in sequential Monte Carlo. *Proceeding of the IEEE*, 95, 899-924.

[17] Mandel J. (2007). A Brief Tutorial on the Ensemble Kalman Filter. *University of Colorado at Denver and Health Sciences Center.*

[18] Mackenzie D. (October 2003). Ensemble Kalman Filters Bring Weather Models Up to Date. *SIAM News*, 36(8), 1-4.