ProbIN: Probabilistic Inertial Navigation

Thanh-Le Nguyen
Karlsruhe Institute of Technology
Kaiserstrasse 12
76131 Karlsruhe, Germany
thanh.nguyen2@student.kit.edu

Ying Zhang
Carnegie Mellon University
NASA Research Park Building 23
Moffett Field, CA 95035, USA
joy.zhang@sv.cmu.edu

Martin Griss
Carnegie Mellon University
NASA Research Park Building 23
Moffett Field, CA 95035, USA
martin.griss@sv.cmu.edu

***Abstract*—Numerous applications require accurate personal navigation for environments where neither GPS signals nor infrastructure beacons, such as WiFi, are available. Inertial navigation using low-cost sensors suffers from the noisy readings which leads to drifting errors over time. In this paper, we introduce a novel inertial navigation approach ProbIN using Bayesian probabilistic framework. ProbIN models the inertial navigation problem as a noise channel problem where we want to recover the actual motion/displacement of the user from the noisy sensor readings. Building on the top of dead reckoning, ProbIN learns a statistical model to map the noisy sensor readings to user's displacements instead of using the double integral of the acceleration. ProbIN also builds a statistical model to estimate the *a priori* probability of a user's trajectory pattern. Combining the mapping model and the trajectory model in a Bayesian framework, ProbIN searches for a trajectory that has the highest probability given the sensor input. Our experiments show that ProbIN significantly reduces the error of inertial navigation using low-cost MEMS sensors in mobile phones.**

***Keywords*-Inertial positioning and navigation, low-cost inertial sensors, Dead Reckoning, Bayes' theorem, Expectation Maximization**

I. INTRODUCTION

Context-aware applications often rely on information about the user's location. To determine the position of the user standardized Global Positioning System (GPS) is utilized outside of the buildings. However, in the environment such as airports, hospitals or nursing homes, GPS signals are unreliable. Even outside GPS signals are not always accurate since they can be blocked by high buildings, canyons or forests [1].

Most of the existing indoor positioning solutions try to address this problem by using pre-installed infrastructure such as network of Wi-Fi access points (APs) [2], [3] or Bluetooth beacons [4]. This works well only when we can assume that there is such infrastructure and it is accessible at the time of the determining the position. While entering a new building complex or in emergency cases (e.g. fire in the building) such assumption cannot be made [5]. When the users walk in a shopping mall they usually do not have an access to the Wi-Fi network. In case of a fire the building can be cut off of the electricity which makes the preinstalled infrastructure useless.

In such scenarios utilizing inertial sensors of mobile devices such as accelerometers, magnetometers and gyroscopes seems to be more appropriate. The movement directions of the user can be tracked by using the actual sensor readings. However, these techniques suffer from the noise generated by the device's sensors. The error of sensor readings accumulates and increases with the time and number of steps.

In this paper we introduce the novel approach which addresses this problem by utilizing the statistical model well-known in the field of the machine translation [6]. We build our approach on an existing inertial positioning system (IPS) which is sufficient for movement within short distances. IPS maps the sensor readings to the actual user's movement. The mapping is used as training data for creating the statistical model. By utilizing this machine learning technique we benefit from the *a priori* knowledge from the collected sensor data instead of applying only basic physics. Thus, the applications can learn the user's common movement and use this information for translating the sensor readings.

II. RELATED WORKS

IPS utilizes sensors such as the accelerometer, magnetometer and gyroscope which are attached to the user's body. The sensor readings are logged continuously in order to determine the user's current position. This is calculated from the previous position and the sensor data using the dead reckoning algorithm [7].

The main drawback of the dead reckoning is the accumulated error, which is caused by the noisy sensor readings. The error increases with the time and traveled distance. The noise can be partially extracted by using a Kalman filter [8] which estimates true values of the observed sensor measurements based on the parameter settings. By utilizing high-quality Inertial Measurement Unit (IMU) containing a combination of accelerometer and ring laser gyroscope the noise can be further reduced [9].

Many research projects utilize pedometer which counts the number of user's step. In combination with the average step length the user's position is estimated. However, these methods are not able to differentiate between different types of movements such as walking, running or jumping. This problem can be addressed by combining dead reckoning, neuronal networks and GPS [7]. The neural network is used to estimate the user's step length. GPS coordinates are used as the reference data to train the neural network.

Other studies use double integral of the accelerometer sensor readings in order to estimate the user's position [10]. For higher accuracy the inertial sensors are attached to the user's body parts such as on the head, to the waist or to the foot. In the systems with sensors mounted on shoes the Zero Velocity Update (ZUPT) can be applied [1], [11]. ZUPT reduces accelerometer's drift error caused by the double integral of the acceleration and noise in the sensor readings. By detecting the stance phase of a step the velocity is reset. Thus, the error made during position estimation in one step will not be carried to the estimation in the next step.

Dead reckoning can be also combined with other techniques in order to achieve higher accuracy. Renaudin et al. [12] described a hybrid system using dead reckoning with RFID for the emergency scenario such as a rescue mission in a burning building. The first two fire fighters enter the building and attach a RFID each time they pass a door. The position of the RFID is estimated from the calculated current position of the fire fighters and the position of the nearest door on the map. Using the RFID the position of the fire fighters is corrected. This system works under the assumption that a database of door positions is available. A similar approach using ultrasound beacons instead of RFID is presented in [13].

Besides the pedestrian navigation, dead reckoning techniques have been used for tracking of wild animals [14]. Shiomi et al. [15] introduced an approach to reconstruct the dive paths of turtles and penguins using accelerometer and magnetometer sensor readings. Instead of using gyroscope the speedometer provides additional information for estimating the orientation. In case the animal travels with a constant speed, the measured acceleration corresponds to the gravity force which is showing down to the earth. Based on this information and the magnetometer sensor readings the orientation of the animal can be calculated.

There are several commercial applications offering indoor positioning for mobile phone devices. Micello.net and PointInside.com provides a floor-plan database of shopping malls, college campuses and stadiums. GPS is used for the indoor positioning. In case that GPS signals are too weak the user is asked to enter the position manually.

In our research we focus on utilizing low-cost inertial sensors in mobile phone devices instead of using expensive special gadgets. Thus, our system is more affordable for the daily usage. The mobile phone device is usually held in the hands or in the pocket. Therefore, it cannot be mounted on shoes or helmets.

III. Probabilistic Inertial Navigation

Probabilistic Inertial Navigation (ProbIN) is a statistical dead reckoning approach. ProbIN frames the "navigation from noisy sensor" problem as a noisy-channel problem [16] where we try to recover the actual position of the user from the distorted sensor input. The sensor readings are noisy by nature. Even with extensive calibration the results are still unsatisfactory. Instead of calculating displacements by double integrating the acceleration, ProbIN considers the sensor information as observed indicators of the underlining motion, i.e., the sensor reading is only used for identifying the user's movement, which is later on mapped to the displacement using our statistical model.

For example, in one case the double integral of an acceleration a over the period of time Δt corresponds to a displacement of $25cm$. However, the actual travelled displacement is $30cm$. By applying the double integral the noise would cause error in the estimation of user's position. By applying the ProbIN approach the model simply learns that sensor reading a corresponds to displacement $30cm$. In another case, the same sensor reading a actually corresponds to a displacement of $28cm$. ProbIN updates the model of mapping a into $30cm$ with probability $P(a|30cm)$ and into $28cm$ with another probability. After collecting enough training data we can estimate the likelihood of different displacements given observed sensor readings. ProbIN assumes that sensor readings are somewhat *consistent* even though they may not be *accurate*, i.e., when a user conducts two similar movements the sensor readings should also be similar.

A. Noisy-channel Model

Noisy-channel models have been widely used in fields such as Statistical Machine Translation (SMT) [6], Automatic Speech Recognition (ASR) and Optical Character Recognition (OCR). The noisy-channel describes the communication process in which the source sends a message through a noisy channel and the target receives a corrupted or ambiguous message. In the case of SMT the aim is to find a translation (source message) that is the most likely source of the observed target message. This issue is addressed by applying Bayes' theorem, which utilizes both the *likelihood* of translating a target language word to source language (translation model) and also *a priori* knowledge about the source languages, namely, a language model for estimating the probability of a source sentence to be a "proper" sentence.

The ambiguity of natural languages is a common problem for SMT. For example, a word in German (e.g. morgen) can have several possible translations in English (e.g., tomorrow, morning, etc.). The German word can be correctly translated only when we know the sentence in which the word is used. Given a German sentence $G = g_1, g_2, \ldots$, we try to find the best English translation $E^* = e_1, e_2, \dots$ as:

$$E^* = \arg\max_E P(E|G) = \arg\max_E \frac{P(G|E) \cdot P(E)}{P(G)} \quad (1)$$

Since $P(G)$ is a constant for all different English translations E, it can be dropped:

$$E^* = \arg\max_E P(E|G) = \arg\max_E P(G|E) \cdot P(E) \quad (2)$$

Similarly, ProbIN tries to find the most likely trajectory of the user given the sensor readings.

B. Quantizing Sensor Readings and Displacements

Unlike SMT where the vocabulary of a language has a limited number of words, the number of possible sensor readings and displacements are unlimited. It is impossible to train a reliable model which can translate every sensor readings into displacements in that continuous space. Instead of using raw sensor readings, we use a sequence of motion labels $M = m_1, m_2, \dots, m_N$ to represent the sequence of N sensor readings collected during the user's movement. We quantize the raw sensor readings using the K-means clustering [17] in order to train statistical models in the discrete space. A sensor reading at timestamp t is represented by one motion label m_t. The continuous space of displacements is also quantized as a sequence of displacement labels $D = d_1, d_2, \dots, d_N$.

Figure 1 shows an example of using the K-means clustering to quantize sensor readings to motion labels.

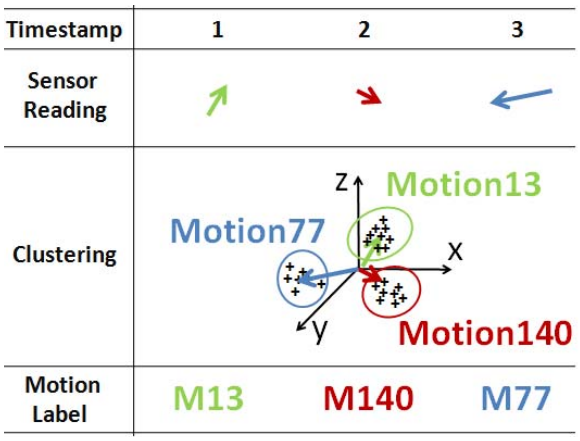

Figure 1. Example of using k-means clustering in order to define the motion labels

For a sequence of motion labels $M = m_1, m_2, \dots, m_n$, ProbIN searches for the optimal displacement labels sequence D^* such that:

$$D^* = \arg\max_D P(D|M) = \arg\max_D P(M|D) \cdot P(D) \quad (3)$$

By concatenating the displacements in D^*, we can estimate the user's ending position at time $t = N$.

C. Translation Model

The statistical model of ProbIN consists of two parts: the Translation (mapping) model and the Trajectory model. Translation model $P(M|D)$ estimates the likelihood of translation/mapping between a sequence of displacement label D and a sequence of motion label M. Assuming the translation of a displacement label to its corresponding motion label is independent of other pairs, we can write:

$$P(M|D) = \prod_{i=1}^{N} P(m_i|d_i) \quad (4)$$

which is the product of probabilities of translating each displacement label d to its corresponding motion label m.

D. Trajectory Model

The *trajectory model* $P(D)$ in Equation 3 is similar to a *language model* used in SMT and Speech Recognition. A language model estimates how likely a sequence of English (or other languages) words is a meaningful English sentence. For example, the sentence "Tomorrow I will go shopping" should have higher language model probability than sentence "Morning I will go shopping" as the former is more likely to be a correct English sentence. In our approach the trajectory model works similarly to the language model. It helps to identify patterns in the user's movement and uses this information for estimating the user's position. For example, when a user is walking forward, his trajectory is most likely to be a sequence of forward moving displacements rather than a sequence of "forward" "backward" "forward" "backward". The trajectory model, trained from the user's past trajectory data (described below), should assign higher probability to "normal" trajectory patterns than those abnormal and thus helps the inertial navigation system to identify a more probable trajectory from all possible alternatives.

The probability of a trajectory D is calculated as:

$$\begin{aligned} P(D) &= \prod_{i=1}^{N} P(d_i|d_1^{i-1}) \quad (5) \\ &= P(d_1) \cdot P(d_2|d_1) \cdot \dots \cdot P(d_i|d_1, \dots, d_{i-1}) \cdot \dots \cdot \\ &\quad P(d_N|d_1, d_2, \dots, d_{N-1}) \end{aligned}$$

Under the Markov assumption that a displacement label d_i only depends on the immediate $n-1$ displacement labels in history, $P(d_i|d_1, \dots, d_{i-1})$ can be estimated as $P(d_i|d_{i-n+1}, \dots, d_{i-1})$. This is equivalent to the n-gram language model approached use in SMT and Speech Recognition.

E. Decoder

Given the input sensor readings which are now quantized as motion labels, the decoder applies the translation and trajectory models to search for the optimal displacement label sequence.

Figure 2 shows an example of decoding a sequence of motion labels. Each hypothesized displacement sequence D is a possible translation of the sequence of the motion labels M with probability $P(M|D) \cdot P(D)$. The hypothesis with the highest probability will be output as a predicted trajectory of the user (red path in Figure 2)

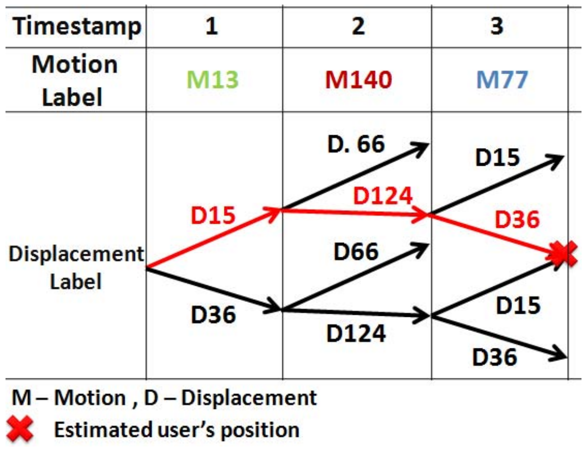

Figure 2. Example of generated hypotheses. The red path corresponds to the hypothesis being most probably the correct translation for the given sequence of motion labels

The complete hypothesis space is exponential to input length N and the number of translation alternatives. If each motion label can be translated into r different displacement labels, the total number of possible trajectories is r^N for an input sequence of N sensor readings. It is computationally infeasible to search the complete hypothesis space for the global optimal trajectory. On the other hand, a greedy search of keeping only one best hypothesis at time t is likely to end up in a local optimum. In order to approximate the globally optimal translation, we use a multi-stack based decoding technique [18]. The multiple stack data structure stores multiple partial hypotheses that have the highest probabilities up to time t as candidates for future expansion. The decoder also applies *hypotheses recombination* technique to merge hypotheses that have the same trajectory model endings and thus can not be distinguished by future hypothesis expansion. For those partial hypotheses that have the same trajectory model endings, only the one with highest probability will survive [19].

The best hypothesis at time N is output as the "optimal" trajectory for the sensor input $m_1, m_2, \ldots, m_N$. Concatenating the displacements along the optimal trajectory leads to the ending position of the user with respect to the starting position.

IV. Training the Statistical Models

To estimate the translation model probability $P(M|D)$ we need a training data where correspondences between motion labels and their displacement label are known. To estimate the trajectory model probability $P(D)$, we need a training data with known trajectories of the user in the format of sequences of displacement labels.

Table I shows an example of collected data used for training the models. Traning data consists of sequences of (motion label, displacement label)-pairs. Data in each sequence is collected during a predefined training process. In Table I the data is collected during several walks. The pairs of (motion label, displacement label) provides information about the correspondences between motion labels and their displacement label. We can additionally extract the sequences of displacement labels from the training data by omitting the motion label in each pair. E.g. we can extract the sequence of (D45, D67, D45, D12, …) from the walk 1. These sequences are used for training the trajectory model.

Walks	Sequence of training data units
Walk 1	(M102, D45), (M102, D67) (M14, D45), (M34, D12) …
Walk 2	(M15, D13), (M102, D45) (M10, D43), (M30, D11) …
…	…

Table I
Example of sequences of training data which consist of (motion label, displacement label) pairs

Based on the observed frequencies of motion/displacement label pairs, the translation probability can be estimated through the Maximum Likelihood Estimation (MLE):

$$P_{MLE}(m|d) = \frac{count((m,d))}{count(d)}. \quad (6)$$

The maximum likelihood estimation of the n-gram trajectory model is based on the displacement label only:

$$P_{MLE}(d_i|d_{i-n+1}, d_{i-n+2}, \ldots, d_{i-1}) = \quad (7)$$

$$\frac{count(d_{i-n+1}, d_{i-n+2}, \ldots, d_{i-1}, d_i)}{count(d_{i-n+1}, d_{i-n+2}, \ldots, d_{i-1})} \quad (8)$$

To avoid the data sparseness problem in the training data, we apply the modified Kneser-Ney smoothing [20] as implemented in the SRI Language Model toolkit [21] .

In practice, obtaining training data such as that illustrated in Table I is infeasible. Getting the accurate displacement for each motion label would require for example high quality motion capture devices. Our work is intended for daily applications where we assume users do not have access to such devices to collect data for training. However, it is reasonable to assume that we can obtain the starting and ending positions either from GPS during outdoor training sessions or from user's input overlayed on floor plans (Table II). Considering the actual displacement label as *hidden* information, we apply the Approximative Expectation Maximization (EM) algorithm [22] to iteratively estimate the hidden displacements for a sequence of motion labels.

EM is an iterative method for finding parameters of a statistical model. Each iteration consists of two steps:

Walks	Sequence of motion labels	Start	End
Walk 1	M102, M102, M14, M34, ...	(0,0)	(4,10)
Walk 2	M15, M102, M10, M30, ...	(4,10)	(7,15)
...	...	...	...

Table II
INFORMATION AVAILABLE FOR TRAINING THE MODELS

Estimation and Maximization. The goal of the *estimation* step is to estimate the hidden displacement using the current statistical models. Based on the current estimation of the hidden displacement information, *maximization* step updates the statistical model for the next iteration.

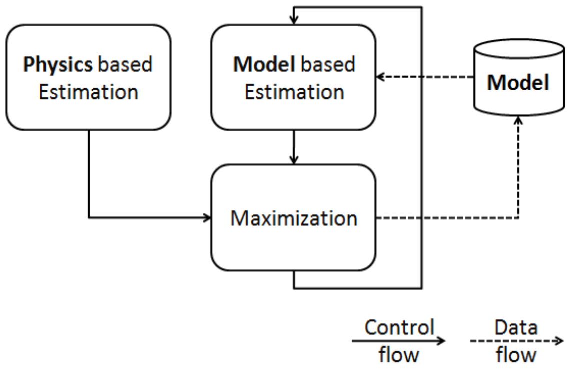

Figure 3. The Approximative EM Algorithm starts with trajectory estimation based on the basic physics and in each iteration updates the statistical mod el

Figure 3 shows the iterative process of the Approximative EM algorithm. A training data contains K instances of "walks":$W_1, W_2, \ldots, W_K$. Each walk W_k has a sequence of motion label M_k, the starting position s_k and the ending position e_k. For each iteration, we use the current model to find the estimated *true* hidden trajectory for each walk W_k. First, we use the current model to decode the motion label sequence M_k to get the estimated trajectory D_k. Most likely D_k is incorrect and does not end at e_k. We assume that the correct trajectory has the same "shape" as D_k, thus we can stretch and rotate D_k such that it ends at e_k. The resulting trajectory D'_k is the newly estimated trajectory for M_k in this iteration. In other words when D_k does not end up at e_k, we know this trajectory is incorrect. Since we don't know which segments caused the error, we distribute the error to each of the N segments in D_k through the stretch and rotation operations.

The stretching factor is calculated as:

$$f_{stretch} = \frac{distance(s_k, e_k)}{distance(s_k, e^*_k)} \tag{9}$$

s_k is the starting position of walk W_k and e_k is the ending position. e^*_k is the ending position based on the estimated trajectory D^*_k. $distance(x, y)$ returns an Euclidean distance between two points.

The rotation angle is calculated as:

$$f_{rotate} = cos^{-1}\left(\frac{e^*_k \circ e_k}{\|e^*_k\| \, \|e_k\|}\right) \tag{10}$$

$\circ$ represents scalar product and $\|x\|$ represents Euclidean norm.

In order to correct the estimated trajectory we correct each d_t by stretching it with the $f_{stretch}$ and rotating it by angle f_{rotate}. The trajectory calculated from the corrected d^{cor}_t should end in the position e_k.

With the corrected estimation of the underlining trajectories D'_k, we can update the translation and trajectory models in order to maximize the probability of the whole training data. The pseudo code of the Approximative EM algorithm is described in Algorithm 1.

```
Data: Training data: a set of $K$ walks:
      $W_1, W_2, \ldots, W_K$. $W_k = (M_k, s_k, e_k)$
iteration ← 0;
while not converge do
    for $k \leftarrow 1$ to $K$ do
        (Estimation step for walk $W_k$) ;
        if iteration==0 then
            Use basic physics (double integral of
            acceleration) to estimate $D^*_k$
        end
        else
            Find $D^*_k$ for $M_k$ given the current statistical
            models
        end
        if $D^*_k$ does not end at $e_k$ then
            stretch and rotate $D^*_k$ to create $D'_k$ such
            that $D'_k$ ends in $e_k$
        end
        else
            $D'_k \leftarrow D^*_k$
        end
        foreach $(m, d)$ pairs in $(M_k, D'_k)$ do increase
        the count of $(m, d)$ ;
        foreach $n$-gram in $D'_k$ do increase the count
        of the trajectory $n$-gram ;
    end
    (Maximization step);
    Update the translation model from counts of $(m, d)$;
    Update the trajectory model from counts of
    trajectory $n$-grams;
    iteration++;
end
```

Algorithm 1: Approximative EM algorithm to train the statistical model in ProbIN.

V. EXPERIMENTS

In our research we focus on the low-cost sensors embedded in mobile phone devices which make the applications more affordable for the daily use. Our approach is evaluated on the Motorola Droid, which is equipped with 3-axis accelerometer and 3-axis magnetometer providing the information about the acceleration in device's coordinate system and the azimuth angle.

Since the device is not equipped with gyroscope the pitch and roll angles cannot be measured. Therefore, during our experiment the mobile device lies on the flat surface of a moving cart (Figure 4). While lying on the flat surface the pitch and roll angles are approximately zero. Thus, no gyroscope is needed.

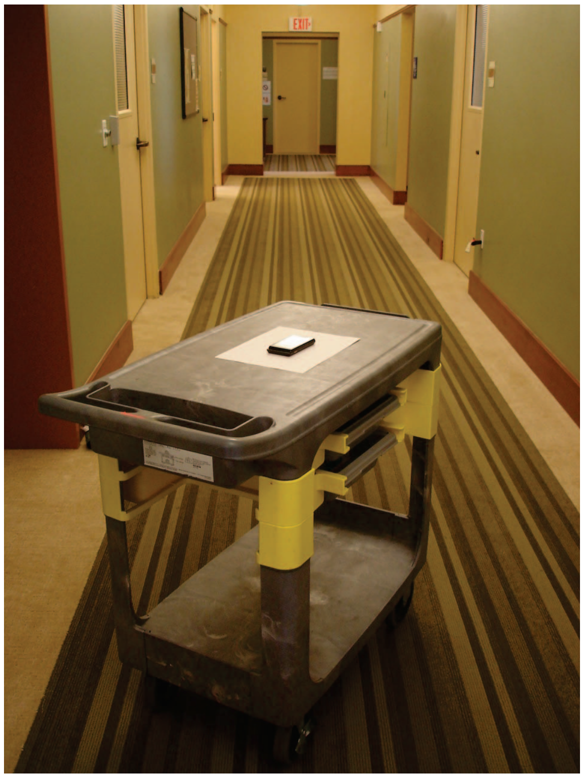
Figure 4. The mobile phone device lies on the cart which is used for our experiment.

In our experiment the user pushes the cart straight forward 40 times for about 10.8 meters in order to simulate the user's walking. The starting position of each walk is the position (0,0) in Figures 5, 6, 7 and 8. The reference end position is in (-10, -4). Sensor readings from 35 walks are used for creating the statistical model and updating it for three EM iterations. The sensor readings from the remaining 5 walks are used for the evaluating the model.

Each sensor readings of the training data set contains acceleration values in the device's coordinate system a_t^{dev}. a_t^{dev} is rotated into the world coordinate system a_t^{world} in order to calculate the traveled displacement. However, instead of using a_t^{world} for training the models we derive training inputs from the acceleration in the device's flat coordinate system a_t^{flat}. This can be obtained by rotating a_t^{dev} only by pitch and roll angle. The azimuth angle is omitted, since it is irrelevant for capturing the characteristics of the user's motion. For example we walk several times straight forward to the north for 10 meters and collect the sensor data. Let's assume that we use a_t^{world} of these walks for training our model. In the decoding phase the model will be able to recognize and translate correctly only the walks to the north. However, if we use a_t^{flat} as training input, walking straight forward to the east, south east or to any other direction can be correctly recognized and translated. By omitting the azimuth angle the training time and effort can be significantly reduced, since much less data will be needed for the training. Correspondingly the training output is derived from corrected displacements in the device's flat coordinate system instead of using the displacements in the world coordinate system.

Since the mobile phone lies on the cart the measured acceleration in device's coordinate system a_t^{dev} corresponds to the acceleration in the device's flat coordinate system a_t^{flat}.

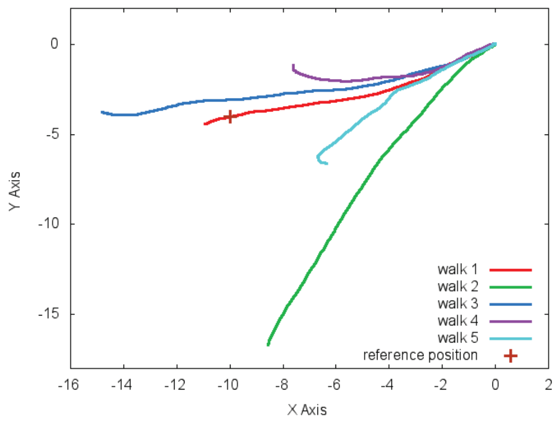

Figure 5. Trajectories calculated by using basic physics

Estimated trajectories using ...	average $error_{relative}$
...basic physics	49.7%
...models from EM iteration 1	31.8%
...models from EM iteration 2	18.3%
...models from EM iteration 3	2.8%

Table III
AVERAGE RELATIVE ERROR OF 5 WALKS CALCULATED USING THE BASIC PHYSICS AND THE MODEL IN EACH EM ITERATION

Figure 5 shows trajectories calculated from the 5 different test walks using dead reckoning approach. Due to the noise in the readings of the low-cost sensors and to the missing gyroscope the estimated end positions of the trajectories differ significantly from the reference end position.

Using the statistical models the result of the estimation is improved. In Figure 6 the trajectories are estimated based

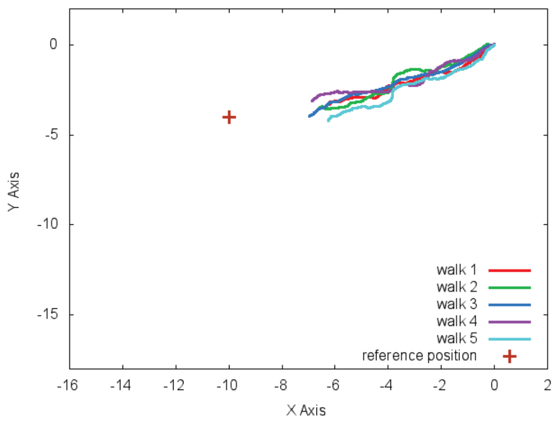

Figure 6. Trajectories estimated by model updated in EM iteration 1

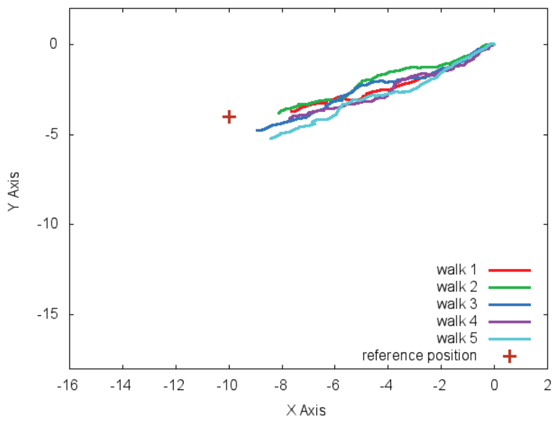

Figure 7. Trajectories estimated by model updated in EM iteration 2

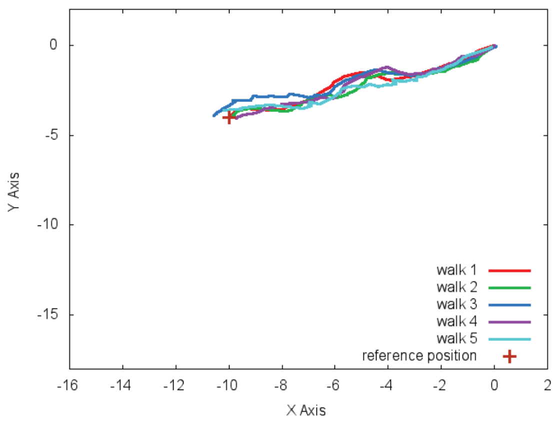

Figure 8. Trajectories estimated by model updated in EM iteration 3

on the statistical models created in the EM iteration 1. The relative error is calculated for each of the 5 walks and the average of the relative errors is shown in the Table III. The error decreases over each iteration and by using the model updated in the EM iteration 3 the average relative error of 2.8% can be achieved.

The relative error is calculated for 2 dimensional trajectories, i.e., the altitude is omitted in our experiment. It is calculated as in the following:

$$error_{relative} = \frac{error_{absolute}}{referenceDistance} \tag{11}$$

The $error_{absolute}$ corresponds to the distance between the user's estimated position p_{est} and the reference position p_{ref}. $referenceDistance$ is the distance between the initial position and the reference end position p_{ref}.

VI. Conclusion

This paper introduces ProbIN, a novel statistical approach of mapping low-cost inertial sensor readings to user's position for indoor navigation applications. Traditional inertial navigation applies the physics model of double integral on acceleration to calculate the displacement. ProbIN learns a statistical model to map the sensor readings to the displacement. The statistical models are bootstrapped using the physics model and are iteratively updated using the EM algorithm. Experiments using low-cost sensors embedded in smart phones shows very promising results compared to standard dead-reckoning with Kalman filters.

We plan to test the ProbIN system on mobile phones equipped with gyroscope such as iPhone 4. Readings from the gyroscope allows the phone to be in any orientation and still map the acceleration from the device space to the absolute space. This allows the phone to provide user's position even when it is held in hand or kept in the pocket. More complex experiments with different type of movements (e.g. running, jumping, etc.) and with longer distances will be conducted. This will allow to use of system in more challenging environments. The calculated error will also include the altitude in order the optimize the system for trajectories taking place on different floors. Our research will be also focus on capturing characteristics of motions general for all users. Thus, we can reduce the training time and effort for each individuals. By using the existing indoor maps the point-to-point navigation will be implemented.

Acknowledgment

Our research is conducted within the interACT Academic Exchange Program between Carnegie Mellon University (USA) and Karlsruhe Institute of Technology (Germany). Special thanks to Google Inc. for providing Motorola Droid for our research. This work is partially sponsored by CyLab and Army Research Office under the contract of W911NF0910273.

References

[1] R. Feliz, E. Zalama, and J. García-Bermejo, "Pedestrian tracking using inertial sensors," *Journal of Physical Agents*, vol. 3, no. 1, pp. 35–43, 2009.

[2] P. Bolliger, "Redpin-adaptive, zero-configuration indoor localization through user collaboration," in *Proceedings of the first ACM international workshop on Mobile entity localization and tracking in GPS-less environments*. ACM, 2008, pp. 55–60.

[3] A. Barry, B. Fisher, and M. Chang, "A Long-Duration Study of User-Trained 802.11 Localization," *Mobile Entity Localization and Tracking in GPS-less*.

[4] S. S. Chawathe, "Low-latency indoor localization using bluetooth beacons," *2009 12th International IEEE Conference on Intelligent Transportation Systems*, pp. 1–7, Oct. 2009.

[5] C. Fischer and H. Gellersen, "Location and Navigation Support for Emergency Responders: A Survey," pp. 38–47, 2010.

[6] P. Brown, J. Cocke, S. Pietra, V. Pietra, F. Jelinek, J. Lafferty, R. Mercer, and P. Roossin, "A statistical approach to machine translation," *Computational linguistics*, vol. 16, no. 2, p. 85, 1990.

[7] S. Beauregard and H. Haas, "Pedestrian dead reckoning: A basis for personal positioning," in *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication (WPNC06)*, 2006, pp. 27–35.

[8] G. Welch and G. Bishop, "An introduction to the Kalman filter," *University of North Carolina at Chapel Hill, Chapel*, pp. 1–16, 1995.

[9] J. Collin, O. Mezentsev, and G, "Indoor positioning system using accelerometry and high accuracy heading sensors," *Proceedings of the 16th*, pp. 1–7, 2003.

[10] X. Yun, E. R. Bachmann, H. Moore, and J. Calusdian, "Self-contained Position Tracking of Human Movement Using Small Inertial/Magnetic Sensor Modules," *Proceedings 2007 IEEE International Conference on Robotics and Automation*, no. April, pp. 2526–2533, Apr. 2007.

[11] L. Ojeda and J. Borenstein, "Non-GPS navigation for security personnel and first responders," *The Journal of Navigation*, vol. 60, no. 3, pp. 391–407, 2007.

[12] B. V. Renaudin, O. Yalak, P. Tomé, and B. Merminod, "Indoor Navigation of Emergency Agents," *Journal of Navigation*, vol. 5, no. 3, 2007.

[13] C. Fischer, K. Muthukrishnan, M. Hazas, and H. Gellersen, "Ultrasound-aided pedestrian dead reckoning for indoor navigation," *Proceedings of the first ACM international workshop on Mobile entity localization and tracking in GPS-less environments - MELT '08*, p. 31, 2008.

[14] M. Johnson, P. Tyack, and W. Instn, "A digital acoustic recording tag for measuring the response of wild marine mammals to sound," *IEEE Journal of Oceanic Engineering*, vol. 28, no. 1, pp. 3–12, Jan. 2003.

[15] K. Shiomi, T. Narazaki, K. Sato, K. Shimatani, N. Arai, P. Ponganis, and N. Miyazaki, "Data-processing artefacts in three-dimensional dive path reconstruction from geomagnetic and acceleration data," *Aquatic Biology*, vol. 8, pp. 289–294, Mar. 2010.

[16] C. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, no. 1, p. 55, 2001.

[17] L. Kaufman and P. Rousseeuw, "Finding groups in data. An introduction to cluster analysis," Mar. 1990.

[18] U. Germann, M. Jahr, K. Knight, D. Marcu, K. Yamada, and Others, "Fast decoding and optimal decoding for machine translation," in *ACL '01: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, vol. 39. Association for Computational Linguistics, 2001, pp. 228–235.

[19] S. Vogel, Y. Zhang, F. Huang, A. Tribble, A. Venugopal, B. Zhao, and A. Waibel, "The CMU statistical machine translation system," in *Proceedings of MT Summit*, vol. 9. Citeseer, 2003.

[20] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, A. Joshi and M. Palmer, Eds. San Francisco: Morgan Kaufmann Publishers, 1996, pp. 310–318.

[21] A. Stolcke, "Srilm - an extensible language modeling toolkit," 2002, pp. 901–904.

[22] A. Dempster, N. Laird, D. Rubin, and Others, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.