Gravity and Linear Acceleration Estimation on Mobile Devices

Samuli Hemminki, Petteri Nurmi, Sasu Tarkoma
Helsinki Institute for Information Technology HIIT
PO Box 68, Department of Computer Science
FI-00014, University of Helsinki, Finland
firstname.lastname@cs.helsinki.fi

ABSTRACT
Linear acceleration is an important enabler for many applications of mobile and wearable activity recognition. The most common approach for estimating linear acceleration is to estimate the gravity component of accelerometer measurements and to project gravity-eliminated accelerometer measurements onto horizontal and vertical planes. Consequently, the accuracy of the linear acceleration estimates is highly dependent on the accuracy and robustness of the underlying gravity estimation algorithm. The present paper contributes by developing a novel approach for gravity and linear acceleration estimation from accelerometer and gyroscope measurements. Our approach improves on previous solutions by (i) providing increased robustness in the presence of sustained acceleration; (ii) detecting and filtering out common types of noise, such as centripetal forces and shifts in device orientation caused by spontaneous user interactions; (iii) operating on shorter time windows, making our approach suitable for applications that require rapid updates on user activities; and by (iv) distinguishing between lateral and longitudinal components of linear acceleration. Experiments carried out using over 100 hours of measurements demonstrate that our approach results in significant improvements in the accuracy of linear acceleration estimates, and improves robustness against common sources of noise in the estimation process. Specifically, our method achieves over 40% improvements in the accuracy of reconstructing speed information and over 70% improvements in the accuracy of estimating travel distances.

ACM Classification Keywords
I.5.2 Pattern Recognition: Design Methodology: Feature evaluation and selection; I.5.4 Pattern Recognition: Applications: Signal processing

General Terms
Algorithms, Experimentation

MOBIQUITOUS 2014, December 02-05, London, Great Britain
Copyright © 2014 ICST 978-1-63190-039-6
DOI 10.4108/icst.mobiquitous.2014.258034

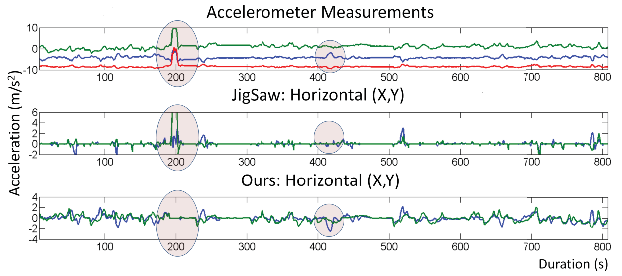


Figure 1. Comparison of linear acceleration estimates between JigSaw (middle) and our approach (bottom). The actual measurements, collected during a commute with a bus, are shown at the top. The highlighted periods correspond to an orientation change (left) and the period of highest acceleration (right).

Author Keywords
Mobile Sensing, Linear Acceleration, Gravity Estimation

INTRODUCTION
Linear acceleration is an important enabler for many applications of wearable and mobile activity recognition as it allows distinguishing actual motion from the effects of gravity. This in turn can be used, e.g., for achieving orientation independence in activity recognition [5, 9], to extract information about vehicular movement patterns [7], to distinguish between everyday transportation modalities [5, 17], to support inertial navigation for autonomous robots moving along uneven terrain [12], and to monitor injury risks [11].

Linear acceleration is typically estimated by deriving an estimate of the direction of the gravity vector and projecting gravity-eliminated accelerometer measurements onto horizontal and vertical planes. Consequently, the accuracy of the linear acceleration estimates depends on the robustness and accuracy of the underlying gravity estimation algorithm. Current solutions to gravity estimation are insufficient for deriving accurate linear acceleration estimates for several reasons. First, they assume gravity to be the only consistent force exerting the accelerometer, thus ignoring sustained acceleration which is frequently present, e.g., during mechanized motion. Second, the approaches ignore common sources of noise, such as centripetal forces exerting the sensor during turns and changes in device orientation, e.g., due to spontaneous user interactions. Third, current gravity estimation algorithms typically require relatively long data windows, e.g., the JigSaw system uses a 4 second window [9] whereas Mizell uses a 30 second window [13], making these approaches unsuitable for real-time sensing applications. The severity of these problems is illustrated in Fig. 1, which compares linear acceleration estimates between the JigSaw system (middle) and our approach (bottom). As measurements we have used data collected on a Samsung S3 smartphone during a bus commute – the original measurements are shown at the top of the figure. From the figure we can observe that JigSaw follows the sensor measurements too closely, removing most of the motion information and being highly sensitive to noise in the measurements. In the figure we have highlighted two sections where these issues are particularly pronounced. The period on the left corresponds to a change in device orientation, which is falsely detected as a period of acceleration. The period on the right contains the period of highest acceleration, which in turn is completely removed from the linear acceleration. In contrast, our approach preserves all information about actual motion, while at the same time filtering out noise and the effects of orientation change.

The present paper contributes by developing a novel approach for gravity and linear acceleration estimation. Our approach operates on a combination of accelerometer and gyroscope measurements, enabling us to decouple rotation and movement, which in turn helps to mitigate the effects of different sources of noise. Our algorithms improve on previous solutions by (i) providing increased robustness in the presence of sustained acceleration or centripetal force; (ii) detecting and filtering out common types of noise, such as shifts in device orientation caused by spontaneous user interactions; (iii) operating on significantly shorter time windows, making our approach suitable for applications that require rapid updates on user activities; and by (iv) enabling the separation of lateral and longitudinal components of linear acceleration. Experiments carried out using over 100 hours of measurements demonstrate significant improvements in the accuracy of linear acceleration estimates. Specifically, we demonstrate over 40% improvements in the accuracy of reconstructing velocity information and over 70% improvements in estimating travel distances using motion sensors. The energy consumption of accelerometers and gyroscopes is relatively low (in total around 35 mW), resulting in a small energy footprint for our approach. Our approach is also computationally lightweight, making it well-suited for mobile devices.

RELATED WORK

Estimating the direction of the gravity vector is a subproblem of the more general problem of *attitude estimation*, i.e., determining the orientation of an object. Attitude estimation has been widely studied, e.g., in aerospace applications, robotics, and motion capture [4, 10, 15]. The main principle is to combine information from two or more complementary sensors to control for different sources of noise. Examples of relevant techniques include variants of IMU-based Kalman filters and complementary filtering techniques. In contrast to mobile devices which rely on lower quality sensors[1], these approaches typically utilize high precision inertial sensing units composed of accelerometers, gyroscopes, and often also magnetometers. Moreover, these approaches assume the effects of noise to be consists over time, making them unsuitable to mobile domains where the effects of noise are dynamic.

One of the first approaches for estimating the gravity component on mobile devices was proposed by Mizell [13] who estimated the gravity component using the mean of accelerometer measurements over a 30 second window. This simple approach generally works fine during pedestrian motion as during these periods the variance in movements is uncorrelated over time, resulting in gravity being the dominant force exerting the accelerometer. Many mobile activity recognition systems have used variations that rely on shorter time windows. For example, Nericell [14] uses the median of a 10 second window, JigSaw [9] uses mean of a 4 second window, and the transportation mode detection system of Wang et al. [17] uses mean of a 8 second window. Generally the longer the window, the better the accuracy, but the higher the latency in the system. However, as illustrated in Fig. 1, these approaches break during sustained acceleration. In addition, these approaches ignore the effects of both centripetal forces and changes in device orientation. An alternative is to use a low-pass filter to extract the gravity component from the measurements. Similarly to the approach of Mizell and its variants, this approach is sensitive to the window length, either failing during sustained acceleration with short window length, or inducing high latency with long window length. The Gravity and Linear Acceleration virtual sensors provided by Android API[2] are based on a combination of these techniques, thus also suffering from the same limitations. In particular, while the virtual sensors work reasonably well during pedestrian movement, they break during mechanized movement that is characterized by sustained acceleration, orientation changes, and centripetal forces.

Kunze et al. [8] estimate gravity opportunistically from accelerometer measurements whenever the variance of the measurements is small and their magnitude approximates the gravity vector. Hemminki et al. [5] propose an extension that adapts the variance threshold over time. This ensures gravity can be estimated when the variance of the measurements is consistently high. The authors also provide a method for detecting orientation changes by comparing the magnitude of the current measurements against the estimated gravity. While these approaches improve the overall accuracy of gravity estimation, they still suffer from some limitations. The main issue is that these approaches are sensitive to sustained acceleration, triggering erroneous updates of the gravity component when the target object is changing velocity or turning.

GRAVITY AND LINEAR ACCELERATION ESTIMATION

Our approach for estimating linear acceleration builds on a novel gravity estimation algorithm that has been designed to mitigate the effects of common sources of inertial noise, such as sustained acceleration, sporadic user interactions, and centripetal forces caused by gyration. Our approach combines

[1] Blum et al. [2] report drifts of up to 4 degrees for smartphone gyroscopes, whereas drifts of dedicated IMUs are below 0.1 degrees.

[2] `http://developer.android.com/guide/topics/sensors/sensors_motion.html`

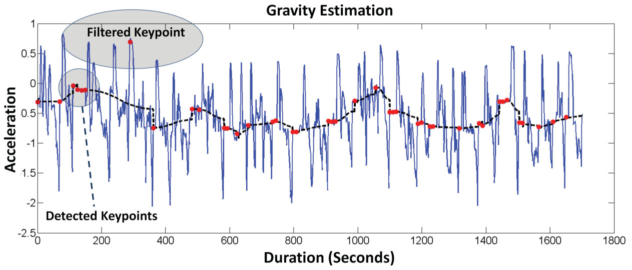


Figure 2. Example of our gravity estimation approach during a tram ride. The solid blue line represents accelerometer measurements along the most informative accelerometer axis. The red dots mark keypoints detected from the measurements, and the black dotted line marks the final estimated gravity.

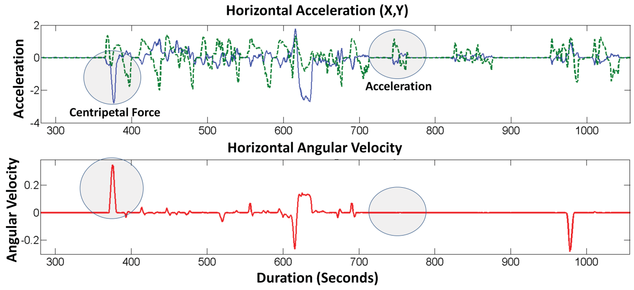


Figure 3. Processed Accelerometer measurements along horizontal plane, and horizontal Angular velocity processed from Gyroscope. Highlighted on the left is a turning event with centripetal force and on the right acceleration event.

accelerometer and gyroscope measurements, which enables decoupling acceleration from device rotation. This in turn provides improved robustness against inertial noise and enables more accurate gravity and linear acceleration estimation. Our approach operates with short measurement frames (one second), which makes it well suited for applications that require real-time motion related information.

Overview

The basic idea in our approach is to detect so-called *keypoints*, which correspond to local minima in the acceleration affecting the user. During these periods, the user is typically either approximately stationary or moving with a constant speed, making gravity the main force exerting the accelerometer measurements. For each detected keypoint, we assign a *stability score* that reflects the overall variability of the measurements around the keypoint. From the keypoints, we derive an estimate of the gravity component by interpolating between the detected keypoints. However, as some of the keypoints can be erroneous, a separate step is used to validate and prune the keypoints before the gravity estimates are interpolated. We also separately monitor for changes in the device orientation and reinitialize the gravity estimation process whenever a substantial change is detected. Once the gravity component has been estimated, we rotate gravity eliminated accelerometer measurements onto global reference plane to obtain an estimate of the overall linear acceleration. Finally, we use correlations between relative changes in device heading and horizontal acceleration to separate between lateral and longitudinal components of the linear acceleration. The overall process for estimating the gravity component and linear acceleration is illustrated in Figures 2 and 3.

Stability of Measurement Frames

The first step in our approach is to detect keypoints that correspond to local minima in the acceleration affecting the user. A naïve approach for this task is to consider points where the variance is sufficiently small as keypoints, as has been done in previous works [1, 8]. While this approach works well during most pedestrian movement, gyration and sustained acceleration, particularly during motorized movement, can result in significant errors in the resulting keypoints. These errors could potentially result in all movement related information being removed from the measurements, as illustrated in Fig. 1. To improve the robustness of the keypoint detection, we combine accelerometer and gyroscope measurements to assign a *stability score* that reflects the overall variability of the measurements within a specific frame. Only frames with sufficiently high stability are considered as keypoints.

Let X denote an arbitrary measurement frame. We define the stability score of the frame as follows:

$$\texttt{StabilityScore}(X) = \alpha_\sigma \sigma_X + \alpha_\mu \Delta\mu_X + \alpha_\omega \omega_X. \quad (1)$$

Here σ_X denotes the standard deviation of the measurements, $\Delta\mu_X$ is the difference in mean between subsequent measurement windows X_i and X_{i-1}, and ω is the magnitude of angular velocity. The difference in mean provides a mechanism for detecting sustained acceleration, whereas angular velocity enables accounting for periods where the sensors are influenced by centripetal forces. Note that the stability score measures overall variability of the measurement frame, and hence a low score implies high stability for a keypoint. The variables α_σ, α_μ, and α_ω are weight terms, which we require to sum up to one. In the experiments we consider $\alpha_\sigma = 0.1$, $\alpha_\mu = 0.45$ and $\alpha_\omega = 0.45$ as the values of the weighting coefficients, i.e., we assume gyration and magnitude changes are the most important factors in determining stability.

Detecting Keypoints

In general, the smaller the variability of the measurements within a frame (i.e., the lower the score), the higher the stability of the frame and the better candidate the frame is for estimating gravity. In practice, however, the stability scores cannot be directly compared as they are sensitive to the overall magnitude of inertial noise, which in turn depends on the current inertial environment of the user. To identify best possible keypoints, we compare the stability of a frame against an adaptive threshold that is adjusted according to the current inertial environment, ensuring the quality of the gravity estimate is the best possible for the given environment.

Our algorithm for detecting keypoints is summarized in Algorithm 1. We compare the stability score S_i (Eq. 1) of the current measurement frame i against an adaptively chosen threshold ϵ_s (Lines 1-2). Whenever the score is within the threshold, the variability in the measurements is small, making the frame a good candidate for gravity estimation. In this

Algorithm 1 `FindKeypoints` $(Gyro_i, Accel_i)$

1: $S_i = \texttt{StabilityScore}(Gyro_i, Accel_i)$
2: **if** $S_i \leq \epsilon_s$ **then**
3: $\quad K_i^S = S_i$
4: $\quad K_i^G = mean(Accel_i)$
5: $\quad keyPoints.add(K_i)$
6: $\quad \epsilon_s = S_i$
7: $\quad Increase = S_i * \eta$
8: **else**
9: $\quad \epsilon_s = \epsilon_s + Increase$
10: **end if**

case, we create a new keypoint K_i that consists of a stability score K_i^S and a gravity estimate K_i^G. As the stability of the keypoint K_i^S we use the score of the corresponding measurement frame (Line 3), and as the gravity estimate K_i^G we consider the mean of the accelerometer measurements within the frame (Line 4). When the stability score of the frame exceeds the threshold, the value of ϵ_s is increased by a value that depends on the stability of the most recent keypoint and a (small) step-size parameter η (Line 9). In the experiments we use $\eta = 0.01$[3]. The threshold is increased until some frame finally meets it, after which the threshold is reinitialized to the stability of the current frame. Thus, the algorithm effectively searches for the lowest possible score value for the current inertial environment, while at the same time ensuring some keypoint candidates are being generated.

Keypoint Validation

In certain situations, the stability of measurements can remain high despite the user being in motion. During these periods, the detection algorithm of previous section would result in an erroneous keypoint being detected. If this keypoint was considered during the the gravity component estimation, all motion related information would be lost. The two most common causes for this problem are illustrated in Fig. 4. On the left-hand side, the user is performing a gentle turning motion during high-speed driving. During this period gyration is relatively low and centripetal forces are approximately constant, resulting in high stability for the measurements. The right-hand figure illustrates the case where the acceleration experienced by the user is approximately constant, resulting again in high stability. However, if keypoints generated from these periods would be used, the resulting gravity estimate would follow the actual sensor measurements too closely instead of preserving the relevant movement related information.

To reduce errors caused by periods of approximately constant acceleration/deceleration or gyration, we perform a separate validation step on the detected measurements before using them for gravity estimation. The basic intuition is to compare whether the accelerometer and the gyroscope detect the same physical phenomena. Specifically, we assume that whenever accelerometer measurements report a change in device inclination, a change of approximately same magnitude should be

[3]This corresponds to a Robbins-Monro stochastic approximation algorithm under the assumption of non-stationarity.

Algorithm 2 `ValidateKeyPoints` $(Gyro\ Accel\ K_i)$

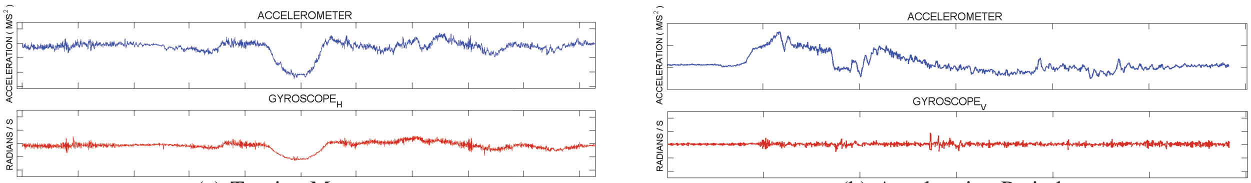
1: $eventRange = findPeakLimits(K_i, Accel)$
2: $R' = rotateData(Gyro(eventRange))$
3: $\Delta\theta_a = getTilt(Accel(eventRange))$
4: $\Delta\theta_r = \sum(R'_{\beta,\gamma}(eventRange))$
5: **if** $|\Delta\theta_a - \Delta\theta_r| > \epsilon_{Acc}$ **then**
6: $\quad Remove\ K_i$
7: **end if**

observed by the gyroscope. In practice, limited sensor quality and the complexity of the user's inertial environment always cause some discrepancies in the measurements. To account for these factors, instead of requiring an equal change, we define an error range ϵ_{Acc}, which defines the acceptable margin of error for the difference between the two inclination changes. If the threshold is exceeded, the keypoint is considered instable and removed from the final gravity estimation.

The keypoint validation algorithm is described in Alg. 2. We first analyze the accelerometer measurements around the keypoint and identify possible peaks in the measurements (Line 1). In case a peak is detected, we calculate the change in inclination separately from those accelerometer and gyroscope measurements that fall within the peak (Lines 3 and 4). For calculating the inclination change from gyroscope, we first need to rotate the measurements onto the reference frame using the gravity estimate K_i^G associated with the keypoint (Line 2). When no peaks are detected, the stability of the measurements is consistently high and the keypoint can be accepted without further processing. If the difference between the two inclination change estimates is larger than the threshold ϵ_{Acc}, the keypoint is rejected. Currently we use $\epsilon_{Acc} = 5°$ as the maximal allowed discrepancy for the sensor readings. The value was selected by finding a maximum difference in sensor measurements, considering the sensor quality and noise in measurements. We opted for a strict threshold as the loss of few true keypoints has low impact on overall gravity estimation accuracy, whereas false keypoints can potentially produce large errors.

Orientation Change Detection

Handling events where device orientation changes substantially is a central challenge in gravity estimation. If not treated properly, these situations can induce a substantial error to the estimated gravity component while the algorithm seeks to converge to the new device orientation. A popular method for this problem has been to monitor the difference between gravity estimates and current accelerometer measurements, triggering orientation change when the difference exceeds a predefined threshold [5, 9]. This method, however, suffers from an inherent flaw in that the accelerometer measurements are influenced by device rotation and acceleration due to motion. Consequently, these approaches can confuse between actual periods of acceleration and changes in device orientation. An alternative is to use device inclination to estimate changes in device orientation. Park et al. [16] demonstrated that inclination is effective at discriminating between different device placements. However, similarly to the approaches using mag-



(a) Turning Maneuver (b) Acceleration Period

Figure 4. Accelerometer measurements and rotated gyroscope measurements during (a) a turn maneuver and (b) a period of sustained acceleration.

nitude for orientation change detection, the approach of Park et al. is sensitive to device rotations and acceleration due to motion as it derives the inclination information directly from accelerometer measurements.

Algorithm 3 `OrientationChange` $(G, Gyro, Accel)$

1: $M = getMagnitude(Gyro)$
2: **if** $M > \sigma_{Event}$ **then**
3: $[\beta, \gamma] = getEulers(G)$
4: $[\beta_{i-n}, \gamma_{i-n}] = getEulers(G_{i-n})$
5: $\Delta\beta = abs(\beta - \beta_{i-n})$ ▷ Pitch
6: $\Delta\gamma = abs(\gamma - \gamma_{i-n})$ ▷ Roll
7: **if** $(\Delta\beta + \Delta\gamma) > \epsilon_A$ **then**
8: **do**
9: $TrackOrientation()$
10: **while** $M > \sigma_{Event}$
11: **end if**
12: **else**
13: $\sigma_{Event} = 3\sigma(M_{1\ldots n})$
14: **end if**

To overcome the problems of previous approaches, we consider a stream-based event detection algorithm that uses gyroscope and accelerometer to monitor for significant shifts in the inclination of the device. The incorporation of gyroscope enables separating device rotation from motion-related information, and the use of an event detection algorithm enables our approach to adapt the detection to the prevailing inertial environment. Our algorithm for detecting changes in orientation is summarized in Algorithm 3. We consider a two-stage detection approach. In the first stage, we use a stream-based event detection algorithm on the magnitude of gyroscope measurements. The event detection algorithm compares the current change in gyroscope magnitude against a threshold σ_{Event} that is determined dynamically based on a running estimate of the standard deviation of gyroscope magnitude (Lines 1 - 2). Currently we set the threshold σ_{Event} equal to three times the standard deviation of recent gyroscope measurements, which effectively corresponds to performing a two-sided t-test with a significance value of 0.0001.

Changes in angular velocity do not directly translate to shift in device orientation as, e.g., a bump in the road or the user shifting position can produce substantial changes in the magnitude of angular velocity. However, in these situations the movement is often symmetric in the sense that the device returns to approximately initial orientation. To avoid triggering orientation change in these events, the second stage of our approach uses changes in device inclination to determine whether the change was due to shift in orientation. The change in inclination, denoted $\Delta\theta$, is calculated as the sum of the changes in the roll γ and pitch β angles, i.e.,

$$\Delta\theta = \Delta\gamma + \Delta\beta. \tag{2}$$

The roll and pitch angles are determined by mapping the current gravity estimate into an Euler angle representation of the device. This is a standard operation related to estimating orientation changes and accomplished using the *getEulers* helper method. When the change in inclination exceeds a threshold ϵ_A (Line 7), we assume the orientation of the device is changing substantially and trigger orientation tracking, which ensures that the orientation change is not interpreted as linear acceleration. We track the orientation using the method proposed by Madgwick [10]. Once the orientation is no longer changing, we store the current orientation and reset the gravity estimate. This ensures the estimated gravity is not continually changing while the user is interacting with the device, and that the gravity estimate rapidly adapts against changes in orientation. Currently we use $\epsilon_A = 10°$ as the threshold.

Gravity Estimation

The frequency of detecting keypoints depends on the user's inertial environment. In some situations, such as traveling by car along an uneven road, there could be a significant delay between successive keypoints. To ensure applications can continuously access high quality gravity and linear acceleration estimates, we use a complementary filter to interpolate gravity estimates in between keypoints. In the *propagation step* of the filter, accelerometer and gyroscope measurements are used to update the gravity estimate derived from the preceding keypoint K_a. Once a new keypoint K_b is detected, a *filtering step* is performed to adjust the estimates between the bounding keypoints K_a and K_b.

Propagation Step

The propagation step is responsible for deriving real-time estimates of the gravity component. The estimates are derived by updating the gravity estimate of the latest keypoint K_a using a combination of gyroscope and accelerometer measurements. The basic form of the progation equation is given by

$$G(x) = (1 - \alpha)G(x - 1) * \omega + \alpha Accel(x) \tag{3}$$

where α is a weight term (we use the default value 0.01) and ω is the rotation matrix derived from gyroscope measurements.

In practice there are two important limitations to this basic scheme. First, gyroscope measurements are inaccurate in terms of drift and noise. Second, the accelerometer measures simultaneously multiple sources of acceleration, making it difficult to decouple the desired effects from noise. To reduce the effects of the different sources of noise, we use an adaptive scheme where the weight of the correction factor α is adapted on the basis of relative frame stability. Specifically, we calculate the correction factor[4] using:

$$\alpha' = \alpha * \frac{K_a^S}{\texttt{StabilityScore}(X_i)}. \quad (4)$$

This scheme allows the interpolation to rapidly correct for drift when the device is in a stable state, while at the same time reducing loss of motion related information during periods of true linear acceleration.

Filtering Step

The stability of the detected keypoints varies depending on the user's inertial enviroment and the quality of the gravity estimates can be further improved by assigning more weight to keypoints with high stability. Whenever a new keypoint becomes available, we use a bi-directional interpolation scheme to adjust the estimated gravity. The use of a bi-directional scheme helps to mitigate drift by reducing the distance between interpolated values and gravity estimates. We derive two gravity estimates using the sequences $G_a = G(x_{1\ldots n})$ (forward interpolation) and $G_b = G(x_{n\ldots 1})$ (backward interpolation). The quality of these interpolations can then be assessed by comparing them with the gravity estimates of the corresponding keypoints, i.e., $E_a = G_a(x_n) - K_b^G$ and $E_b = G_b(x_1) - K_a^G$. The final gravity estimate is calculated as a weighted combination of the two interpolations, assigning higher weight to interpolation with a small error. We use a non-linear (Sigmoid) weight function, given by:

$$\begin{cases} W = 0 \,, i = 1 \\ W = 2(\frac{x}{n})^2 \,, 1 < i < c \\ W = 1 - 2(\frac{x-n}{n})^2 \,, c < i < n \\ W = 1, i = n. \end{cases} \quad (5)$$

Here n is the number of measurements between keypoints and $c = \frac{nE_a}{E_a+E_b}$ is a weighting factor determining the relative interpolation quality between the forward and backward estimates. The final interpolation G can now be expressed as:

$$G = (1 - W)G_a(x_{1\ldots n}) + WG_b(x_{n\ldots 1}) \quad (6)$$

Adaptation to Application Requirements

We have also designed an extension of the above mentioned scheme which allows applications to adjust the adaptation mechanism by specifying an "acceptable" range for the quality of the interpolation. In this case, applications specify a range $[\epsilon_{minE}, \epsilon_{maxE}]$ for interpolation quality, and we automatically adjust the accelerometer correction factor to meet the desired quality criterion. The adaptation mechanism monitors the quality of the gravity interpolations, and in case the interpolation quality falls outside the desired range, reiterates with adapted accelerometer correction factor α. Since the interpolation weight W already emphasizes the better of the interpolations G_a, G_b, we use $min(E_a, E_b)$ to define the current interpolation error. For the adaptation we employ a simple mechanism where α is doubled whenever the error is higher than ϵ_{maxE} and halved whenever it is lower than ϵ_{minE}. As the default values we use ϵ_{minE} = 0.01g and ϵ_{maxE} = 0.05g. Accordingly, when the error is low, we reinterpolate with a smaller correction factor alpha to maximize the amount of motion information that is preserved. In the opposite case, the user's inertial environment contains lots of noise. To mitigate the effects of noise, the correction factor is increased to put more weight on the accelerometer values.

[4]The same adaptation scheme is used in the filtering step, with the exception that we use $\min\{K_a^S, K_b^S\}$ as the denominator

Extracting Linear Acceleration

Once the gravity component has been estimated, we decompose the linear acceleration into a three dimensional vector consisting of vertical L_v, longitudinal L_{lo} and lateral L_{la} accelerations. We start by rotating the accelerometer measurements onto the reference frame. We use A_x, A_y and A_z to denote the rotated measurements.

Estimating the linear acceleration along vertical plane L_v is trivial as this can be solved by simply removing the vertical component of the gravity vector from the vertical component of rotated accelerations, i.e.,

$$L_v = A_z - G_z. \quad (7)$$

Lateral and longitudinal linear acceleration, however, are more difficult to solve, since the sensor measurements do not directly include information about the direction of movement on the horizontal plane. We approach solving L_{la} and L_{lo} by first defining two orthogonal unit vectors, $\hat{u} \perp \hat{v}$ pointing to lateral and longitudinal directions on the horizontal plane. Next, we define key periods, $Accelerating$, $Breaking$ and $Turning$, during which the sensor measurements contain most information about direction of horizontal motion. $Accelerating$ is triggered when there is a period of acceleration following a stopping period. Similarly, $Breaking$ is detected when there is a period of acceleration preceding a stopping period. During these periods, the direction of movement on horizontal plane can be estimated using four-quadrant inverse tangent $\hat{v} = \arctan(A_x, A_y)$. $Turning$ is detected by inspecting the correlation ρ between $R_z^{'}$ and $A_{x,y}$, i.e., between rotation around vertical plane and acceleration on horizontal plane. Correlation indicates presence of centripetal force, which can be used as reference for lateral direction. We use correlation as a weighting term, and estimate the unit vector $\hat{u}$ pointing in lateral direction using:

$$\rho = corr(R_z^{'}, A_{x,y}) \quad (8)$$

$$\hat{u} = \arctan(\frac{\rho_x A_x}{norm(\rho)}, \frac{\rho_y A_y}{norm(\rho)}). \quad (9)$$

Finally, since $\hat{u}$ and $\hat{v}$ are orthogonal vectors, we can evaluate the reliability of direction estimate by calculating $\hat{v} \cdot \hat{u}$. Ideally, $\hat{v} \cdot \hat{u}$ = 0, which indicates flawless reliability. Any deviation from 0 can be interpreted as a measure of uncertainty about the horizontal direction and can be relayed to receiving application for futher processing.

Since the estimated direction $\hat{u}$ and $\hat{v}$ is only valid while the device orientation remains relatively unchanged, we use detected orientation shifts to segment the data into periods where the orientation remains stable. Only measurements from these periods are used to estimate the correlations.

BENCHMARK EVALUATIONS

We evaluate our techniques through a combination of micro-benchmarks that focus on the different sub-algorithms, and an evaluation of the accuracy of the overall approach by comparing the estimated linear acceleration measurements against GPS ground truth. In the following we describe the micro-benchmark evaluations. The accuracy of linear acceleration estimation is discussed in the subsequent section.

Dataset

Our micro-benchmark evaluations have been carried out using 67 hours of measurements collected from nine individuals. The measurements were collected using two controlled scenarios, which were designed to contain a large variety of everyday transportation modes and traffic conditions. To demonstrate that our results are not specific to the scenarios considered, the evaluation of the overall system was carried out using a separate set of measurements collected from everyday transportation behavior; see the next section.

The two scenarios took place within the downtown area of a medium-sized European city with around $750,000$ inhabitants and extensive public transportation network. Each scenario lasted between 90 and 120 minutes per participant. The first scenario was carried out during the winter of $2011-2012$ and included nine participants. The second scenario was carried out during autumn of 2012 and included seven participants. Each participant carried three phones, placed at the three most common placements for a mobile phone in an urban space [6]: trouser pockets, jacket pockets, and bag. Ground truth annotations were made on an additional mobile phone to avoid disturbing the sensing units. We consider frames consisting of one second of gyroscope and accelerometer measurements. We assume the frames are constructed without overlap. In line with current best practices, a low-pass filter was applied on both sets of measurements to reduce the influence of jitter. In the data collection we have employed five different phone models: Samsung Nexus S, Samsung Galaxy S II, Samsung Galaxy S III, Samsung Galaxy S 4 and Samsung Galaxy S 5. The sampling rates of the sensors varied from 50 Hz to 190 Hz during the experiments.

Keypoint Detection Performance

We first consider the performance of the keypoint detection algorithm by inspecting the conditions where keypoints are generated. To carry out this evaluation, we consider a subset of the measurements collected from controlled scenarios that covers different motorized modalities (bus, car, and tram modalities). We selected motorized modalities that operate amongst other traffic to ensure the data contains more dynamic variations, thus posing a more challenging environment for gravity and linear acceleration estimation. The data contains labels for stationary periods during segments of motorized segments, i.e., periods where the user is in a vehicle, but the vehicle is completely stopped. As these periods represent the best opportunities for keypoint generation, we evaluate the keypoint detector by comparing how well the generated keypoints align with the stationary periods.

Modality	Prec	Rec	S.Periods	Mean Stability
Bus	92	98	186	0.043
Tram	87	97	509	0.026
Car	40	99	115	0.048

Table 1. Keypoint detection performance.

We consider precision and recall as our evaluation measure. The recall of the keypoint detection reflects how often a stationary period induced the keypoint detector to trigger, whereas precision reflects how often the triggered keypoints corresponded to actual stationary periods. As we do not consider the filtering step at this stage of the evaluation, recall is more important than precision as it defines how often the best opportunities are detected.

The results of this evaluation are summarized in Table 1. Out of the 810 stationary periods, keypoints were generated during 97%, i.e., 789 stationary periods, indicating an excellent recall. The few occasions where keypoint was not generated during a stationary period were mainly caused by two events: either (i) the user was moving or interacting with the device, resulting in a low stability, or (ii) there were consequent stationary periods where a high-stability stationary period was closely followed by a low-stability stationary period. In terms of precision, we observe that most of the keypoints were generated during stationary periods for the bus and tram modalities, but only 40% for car modality. This relates to the stopping frequency of the different modalities, since compared to car, bus and tram stop more frequently to let passengers in and out. Inspecting the average stability scores, however, reveals that our method is able to find keypoints also outside the periods explicitly labeled as stationary with only modest effect on keypoint stability.

Keypoint Validation

We evaluate the performance of keypoint validation step using measurements collected in a known device orientation. Since the gravity component G is known for these measurements, any keypoint outside an error corridor G $\pm$ ϵ_G in Alg. 2 can be defined as a false keypoint.

Out of the 941 detected keypoints 211 were filtered, of which 133 were false keypoints and 78 true keypoints. In total 7 false keypoints were left unfiltered, all of which were associated with an acceleration or breaking period. Since removing a few true keypoints is secondary to removing false keypoints, we next focus on the 7 unfiltered false keypoints. Of the 7 false keypoints, 4 were the first keypoint of a segment. The first keypoints are problematic for the validation algorithm if they are detected during the first few frames of the data, since in these situations we are left with incomplete information about peaks in acceleration. The other three unfiltered false keypoints were generated during acceleration or breaking periods in high-speed motorway drive. In these cases, the algorithm fails to detect false keypoints due to the noise caused

Magnitude	Ang.Velocity	Detected
2-5°	fast	0/10
5-20°	fast	9/10
20-90°	fast	10/10
20-60°	slow	9/10
10-20°	slow	7/10

Table 2. Orientation Changes induced and detected.

by vehicle movement to gyroscope measurements. The error caused to gravity estimation by both cases is limited, since the stability of the keypoints generated at start of segment or high speed motorway drive is low, and a new keypoint is typically generated within a few tens of seconds.

Orientation Change Detector

We next demonstrate the effectiveness of our approach for detecting changes in device orientation. We consider two evaluations. First, we show that the algorithm triggers from changes in the inclination θ and that it is able to track the device's orientation during these events. Second, we show that the algorithm does not trigger during periods of strong acceleration or vehicle turning maneuvers.

To evaluate that our orientation change detector is triggered by change in tilt angle θ, we produced 5 sets of 10 orientation changes with varying angular velocity and magnitude. For a summary of the performed orientation changes, see Table. 2. The first three sets of orientation changes were between approximately 2-5°, 5-20° and 20-90° respectively, all performed rapidly within a few seconds. The last two sets of large and intermediate orientation changes were performed gradually, within $5 - 10$ seconds. Overall, the orientation change detector performed as expected. The performance is somewhat lower during gradual orientation changes, but in everyday usage these situations are rare, as the orientation changes tend to be fast (e.g., taking a phone to the hand or orientation changing while getting out from a car).

To demonstrate that the orientation change module is not triggered by vehicular movement, we have performed a controlled experiment where two smartphones were set in a fixed orientation during a car drive; one mounted to fixed position and one kept inside drivers pocket. A series of acceleration, breaking and turning maneuvers of various intensities within limits of reasonable safety were performed to test the robustness of orientation change detector. Maximum change in measurements due to acceleration and breaking was $2.9m/s^2$. For centripetal force, maximum change was $3.2m/s^2$. Maximum angular velocity measured by gyroscope was 39°/sec. No orientation changes were triggered by either of the smartphones, confirming that our orientation change detection can avoid false triggering.

Summary

The results of the micro-benchmark evaluations demonstrate that our methods are capable of detecting periods of relative stability in the user's inertial environment with high precision and recall. Ensuring high stability is a key prerequisite for accurate gravity estimation as it guarantees the estimates are derived from periods where gravity is the dominant force exerting the accelerometer, enabling our approach to better preserve motion related information.

Similarly, the results of the performance evaluation of the orientation change detection algorithm demonstrated accurate performance. The performance was particularly good during periods where the changes in angular velocity are large, as would be the case in most situations of practical interest. Examples of situations that should be recognized based on our results include user picking up the device to interact with it, user standing up, or user sitting down.

LINEAR ACCELERATION ESTIMATION ACCURACY

We next demonstrate the effectiveness of our overall solution in estimating linear acceleration. We consider the most important scenario for linear acceleration, namely that of speed and relative distance estimation through integration and double integration of linear acceleration. As ground truth measurements we consider a combination of GPS measurements and information about the actual route, where available.

Measurements

We carry out the evaluation using 47 hours of measurements collected from everyday transportation. As the measurements were collected in an unsupervised setting, no restrictions to device placement or use were imposed. Ground truth was assessed either through GPS (sampled at 1Hz rate) or information about the taken route, where available. In total, we consider 767 kms of transportation data, out of which 315 km were evaluated based on knowledge about route paths and 452 km based on GPS information.

Baselines

We compare our approach against state-of-art gravity estimation techniques. The Madgwick sensor fusion [10] represents the state-of-the-art baseline of combined gyroscope and accelerometer based systems. We also compare our approach against two accelerometer-based solutions: the method used in the JigSaw Continuous Sensing Engine [9] and the method proposed by Kunze et al. in [8].

Experimental Setup

For GPS based evaluation, we divide each trip to segments using stopping periods obtained from GPS speed as a delimiter. For each segment we calculate the distance and speed from linear acceleration estimates, and compare it with the distance and speed information given by the GPS. For distance, we use absolute distance in meters and relative distance compared to GPS distance. For speed, we use correlation between integration of linear acceleration and speed reported by GPS. Since no speed information during the segments is available, we compare the total distance accumulated by double integration of acceleration with the distance measured from route path.

Results

The overall accuracy of the different methods is summarized in Table 3. From the results we can observe that the most accurate estimates are given by our method, providing a high correlation with GPS speed and resulting in small distance error both for segments and entire routes. The results also show

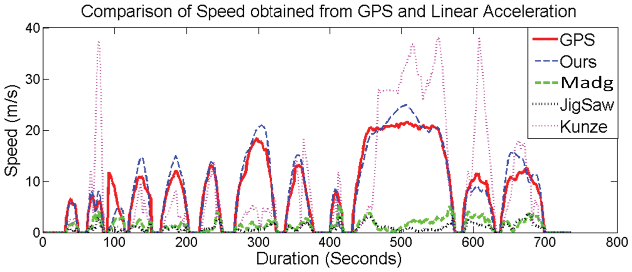


Figure 5. Comparison of GPS speed and speed estimated from linear acceleration during an example bus trip

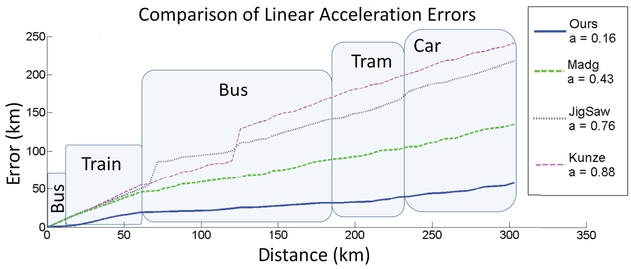


Figure 6. Comparison of the accumulation of distance error of the evaluated approaches, where a notes regression coefficient. The comparison is performed against distance obtained from GPS measurements.

that the combination of accelerometer and gyroscope measurements provides clear performance gains as both our approach and the complementary filtering technique of Madgwick outperform the accelerometer-based baselines. In terms of the accelerometer-based baselines, the method of Kunze et al. is better at capturing changes in velocity, but overestimates them, and consequently also distance. The method used in JigSaw, on the other hand, tends to follow the measurements too closely, underestimating both speed and distance traveled.

To better understand the performance differences, Fig. 5 illustrates the performance of the different methods in estimating the user's velocity during a bus trip. Both Madgwick and the algorithm used in JigSaw follow the accelerometer measurements too closely, resulting in a significant underestimation of the user's true velocity, and hence also distance. The method of Kunze et al. captures changes in velocity more accurately, only failing to recognize gradual changes (see, e.g., at around 300 second mark). However, the method tends to react too slowly to sudden changes, resulting in significant overestimates of velocity. In contrast to the other methods, our approach correctly recognizes all changes in velocity and is capable of tracking the extent of the changes accurately.

To further demonstrate the benefits of our approach, Fig. 6 compares the total accumulated error in distance over the data containing route-based ground truth information. All methods exhibit approximately linear drift on the data, with the error of our approach being consistently smallest. Moreover, the differences are consistent across all of the transportation modalities considered in the evaluation. Especially noteworthy is the small drift during car travel, since the stability of the keypoints tends to be lower during these periods, as illustrated in the micro-benchmark evaluations. We assessed the significance of the differences using linear regression on the accumulated errors and performing a z-test on the differences between coefficients. All methods had extremely good model fit ($R^2 > 0.9$) with our method providing over 60% smaller coefficient (0.16 vs. 0.43) than the second best method (Madgwick). The difference in coefficients between our approach and the complementary filter of Madgwick was found statistically significant ($z = -44.35, p << 0.001$).

DISCUSSION AND SUMMARY

The present paper has contributed by developing a novel approach for gravity and linear acceleration estimation on mobile devices using a combination of accelerometer and gyroscope measurements. Our experiments, conducted using over 100 hours of measurements collected from diverse transportation situations, demonstrate good performance in linear acceleration estimation and robustness against the most common sources of inertial noise. In particular, we have shown that the linear acceleration estimates provided by our approach can be used to accurately estimate both the velocity and traveled distance of the user in a variety of transportation situations. We have also demonstrated that our approach can operate well in everyday mobile computing situations where the orientation of the device is unconstrained and can change dynamically.

Relying solely on inertial sensors entails several benefits for energy-efficiency as the energy consumption of accelerometer and gyroscope is low compared to other device functionalities. Specifically, the power consumption of accelerometers is typically around 15 mW, and the gyroscope consumes approximately 20 mW[5], whereas background processes typically consume around $100 - 150$ mW and the GPS consumes around 175 mW. The power costs are likely to go down in the near future as increasingly many smartphones are integrating accelerometers and gyroscopes on the same chip[6]. In terms of processing time, while requiring more processing than accelerometer-based solutions, the processing time of our approach is negligible in practice and well-suited for real-time applications on mobile devices. On a high-end smartphone, the average processing time of a measurement frame is around 10ms, which is over 55% decrease compared to the complementary filtering technique of Madgwick.

The approach presented in this paper enables capturing linear acceleration accurately, which in turn is a fundamental enabler for many activity recognition tasks [3, 5, 7, 14]. Of particular interest are the many vehicular motion related applications enabled by our work. The linear acceleration could be used for e.g., accurate transportation mode detection, fuel consumption estimation and detection of driving maneuvers. Vehicle movements could also be used as electronic evidence for digital forensics, providing valuable information about accident situations. Also the gravity component provides im-

[5]Some Android variants have been reported to have a significantly higher energy consumptions when high sampling rate (100Hz) is used. However, this is likely to be caused by a system bug rather than the sensor itself.

[6]http://www.chipworks.com/en/technical-competitive-analysis/resources/blog/inside-the-samsung-galaxy-s4/

Ground truth from GPS.

Method	Speed (ρ)	Seg. Err (m)	Seg. Err (%)
Ours	0.84	-75 m	-21%
Madgwick	0.57	-205 m	-68%
JigSaw	0.60	-228 m	-78%
Kunze	0.47	+239 m	+91%

Ground truth from route information.

Method	Route Err (m)	Route Err (%)
Ours	-606 m	-10%
Madgwick	-2034 m	-35%
JigSaw	-4127 m	-71%
Kunze	+4712 m	+83%

Table 3. Comparison of longitudinal linear acceleration with ground truth information.

portant information to multitude of other applications, e.g., tilt correction for magnetometer, device placement detection and adapting the orientation of device display.

ACKNOWLEDGMENTS

The work was supported in part by the DIGILE IoT SHOK programme. The work only reflects the authors' views. The authors are grateful to Patrik Floréen, Sourav Bhattacharya, Teemu Pulkkinen, Joel Pyykkö, Tony Kovanen, Guo Haipeng, Yina Ye, Taneli Vähäkangas, and Kalle Myllymaa for their help in data collection.

REFERENCES

1. Bhattacharya, S., Blunck, H., Kjærgaard, M., and Nurmi, P. Robust and energy-efficient trajectory tracking for mobile devices. *IEEE Transactions on Mobile Computing*, In Press (2014).
2. Blum, J. R., Greencorn, D. G., and Cooperstock, J. R. Smartphone Sensor Reliability for Augmented Reality Applications. In *Proceedings of the 9th International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services (MobiQuitous)* (2012).
3. Gabrielli, S., Forbes, P., Jylhä, A., Wells, S., Siren, M., Hemminki, S., Nurmi, P., Maimone, R., Masthoff, J., Jacucci G. Design challenges in motivating change for sustainable urban mobility. *Computers in Human Behavior*, In Press (2014).
4. Gallagher, A., Matsuoka, Y., and Ang, W.-T. An efficient real-time human posture tracking algorithm using low-cost inertial and magnetic sensors. In *International Conference on Intelligent Robots and Systems (IROS)* (2004).
5. Hemminki, S., Nurmi, P., and Tarkoma, S. Accelerometer-based transportation mode detection on smartphones. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, ACM (2013).
6. Ichikawa, F., Chipchase, J., and Grignani, R. Where's the phone? A study of mobile phone location in public spaces. In *Proceedings of the 2nd International Conference on Mobile Technology, Applications and Systems*, IEEE (2005), 1 – 8.
7. Johnson, D. A., and Trivedi, M. M. Driving style recognition using a smartphone as a sensor platform. In *Proceedings of the 14th International IEEE Conference on Intelligent Transportation Systems (ITS)* (2011).
8. Kunze, K., Lukowicz, P., Partridge, K., and Begole, B. Which way am I facing: Inferring horizontal device orientation from an accelerometer signal. In *Proceedings of the 13th IEEE International Symposium on Wearable Computers (ISWC)* (2009), 149–150.
9. Lu, H., Yang, J., Liu, Z., Lane, N. D., Choudhury, T., Campbell, T. The Jigsaw continuous sensing engine for mobile phone applications. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems* (2010), 71–84.
10. Madgwick, S. O. H., Harrison, A. J. L., and Vaidyanathan, R. Estimation of IMU and MARG orientation using a gradient descent algorithm. In *Proceedings of the IEEE International Conference on Rehabilitation Robotics*, IEEE (2011), 1–7.
11. Marjoux, D., Baumgartner, D., Deck, C., and Willinger, R. Head injury prediction capability of the HIC, HIP, SIMon and ULP criteria. *Accident Analysis & Prevention 40* (2008), 1135 – 1148.
12. Matveev, A. S., Wang, C., and Savkin, A. V. Real-time navigation of mobile robots in problems of border patrolling and avoiding collisions with moving and deforming obstacles. *Robotics and Autonomous Systems 60*, 6 (2012), 769 – 788.
13. Mizell, D. Using gravity to estimate accelerometer orientation. In *Proc. Seventh IEEE International Symposium on Wearable Computers* (18–21 Oct. 2005), 252–253.
14. Mohan, P., Padmanabhan, V. N., and Ramachandran, R. Nericell: Rich monitoring of road and traffic conditions using mobile smartphones. In *Proceedings of the 6th ACM conference on Embedded network sensor systems (SenSys)*, ACM (2008), 323–336.
15. Nguyen, H. Q. P., Kang, H.-J., Suh, Y. S., and Ro, Y. S. A DCM Based Orientation Estimation Algorithm with an Inertial Measurement Unit and a Magnetic Compass. *Journal of Universal Computer Science 15* (2009), 859–876.
16. Park, J., Patel, A., Curtis, D., Ledlie, J., and Teller, S. Online pose classification and walking speed estimation using handheld devices. In *Proceedings of the 14th ACM International Conference on Ubiquitous Computing (Ubicomp)* (2012).
17. Wang, S., Chen, C., and Ma, J. Accelerometer based transportation mode recognition on mobile phones. In *Asia-Pacific Conference on Wearable Computing Systems* (2010), 44–46.