Use It Free: Instantly Knowing Your Phone Attitude

Pengfei Zhou, Mo Li
Nanyang Technological University, Singapore
{pfzhou, limo}@ntu.edu.sg

Guobin Shen
Microsoft Research, China
jackysh@microsoft.com

ABSTRACT

The phone attitude is an essential input to many smartphone applications, which has been known very difficult to accurately estimate especially over long time. Based on in-depth understanding of the nature of the MEMS gyroscope and other IMU sensors commonly equipped on smartphones, we propose A^3 – an accurate and automatic attitude detector for commodity smartphones. A^3 primarily leverages the gyroscope, but intelligently incorporates the accelerometer and magnetometer to select the best sensing capabilities and derive the most accurate attitude estimation. Extensive experimental evaluation on various types of Android smartphones confirms the outstanding performance of A^3. Compared with other existing solutions, A^3 provides $3\times$ improvement on the accuracy of attitude estimation.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Modeling techniques, Performance attributes; C.5.3 [**Computer System Implementation**]: Portable devices

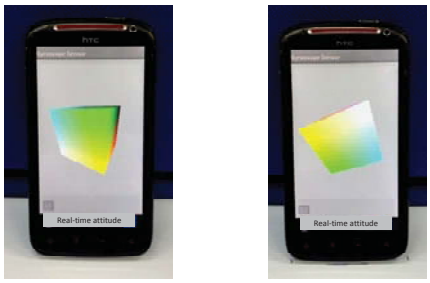
Keywords

Mobile Phone Attitude, Gyroscope, IMU Sensors, Attitude Calibration

1. INTRODUCTION

The phone attitude gives the 3D orientation of the phone with respect to the earth coordinate system. It is an essential input to many phone applications including mobile gaming, 3D photography [12, 18], gesture and activity recognition [17, 20], dead reckoning based localization and navigation [11, 24, 25, 26], etc. Since the first integration into smartphones in 2010, the MEMS gyroscope, capable of measuring the 3 dimensional angular velocities, has been exploited to estimate phone attitude by continuously integrating the angular velocities. However, unlike those high precision gyroscopes used in nautical, aviation and robotic navigation systems, the MEMS gyroscope has been widely blamed for its poor accuracy.

The problem becomes particularly challenging in practice as people may take their phones in arbitrary ways, hold in different body positions such as in hands, pockets, or bags, and make complicated motion styles. Figure 1 shows the phone attitude estimation results of a popular Android app[1] using the gyroscope. Figure 1(a) depicts the initial attitude estimation and Figure 1(b) depicts the estimation for the same phone attitude after 1 minute random motion. Obviously, there is a significant difference in the two attitude estimates, which corresponds to the phone attitude tracking error. Some recent research works exploited the gyroscope for heading direction estimation. Their reports also confirmed the severity of gyroscope drifts in minute level runs even when the mobile phone is held relatively still to the user [19, 22, 24]. Although there have been many efforts made to address a similar problem (rigid body orientation) in robotics domain [14, 15, 8, 16], our study reveals that directly adopting those approaches cannot provide satisfactory performance, mainly due to the lack of fundamental understanding of the MEMS gyroscope and other IMU (Inertial Measurement Unit) sensors on smartphones. As a concrete example, current Android API directly borrows existing Kalman-based sensor fusion techniques to estimate the phone attitude and suffers significant performance degradation in many conditions.



(a) Initial attitude estimation. (b) Attitude estimation after 1 minute.

Figure 1: Phone attitude tracking error of a popular smartphone app.

In this paper, we conduct detailed experimental study to understand the fundamental performance of the MEMS gyroscope. We follow the device specification and conduct controlled experiments to investigate how different environmental factors impact on the gyroscope performance and how the best accuracy can be achieved in an appropriate condition range. We also characterize the nature of two other IMU sensors on smartphones, the accelerometer and compass, to understand their performance in different conditions. Based on such comprehensive understanding, we propose to estimate the phone attitude primarily based on the 3-axis angular ve-

[1]The app is called "Android Sensor Box" available at Google Play with about 1,000,000 downloads.

MobiCom'14, September 7-11, 2014, Maui, Hawaii, USA.

http://dx.doi.org/10.1145/2639108.2639110.

locities obtained from the gyroscope, but also incorporate the compass and the accelerometer for opportunistic calibration. The three types of IMU sensors are of different natures and their accuracy varies in different condition ranges. In particular, the gyroscope provides a cumulative estimation of the attitude through continuous integration on angular velocities that is accurate in general but suffers from error accumulation. On the other hand, the compass sensor and accelerometer provide instant attitude estimation without cumulative errors but the accuracy is highly instant environment and motion dependent. We develop a practical approach that calibrates the cumulative gyroscope estimation when we have higher confidence in compass and accelerometer readings. In practical usage, it is non-trivial to identify "good" calibration opportunities. Intuitively, the compass sensor outputs accurate geomagnetic north when the phone is outdoor and the gravity direction can be accurately extracted from accelerometer when the phone is static. Such opportunities, however, would be too few in practice to provide timely calibration. In this work we propose an "opportunistic calibration" technique that looks at the concordance of the three types of sensors in estimating short period attitude changes. High consistency indicates high instant confidence of the compass and gravity outputs and thus a positive calibration opportunity. Sufficient calibration opportunities can be identified using this approach.

We incorporate the proposed design and techniques and develop A^3, an Accurate and Automatic Attitude detector for smartphones. A prototype system is implemented and comprehensively tested with three types of Android smartphones, including HTC Sensation, Samsung i9100, and LG Google Nexus 4. We evaluate the performance of A^3 across various scenarios and in multiple popular apps, and compare A^3 with other possible competitors. The experiment results demonstrate that A^3 provides $3\times$ improvement over alternative solutions. Higher performance gain can be achieved the user motion is intense (e.g., running). Meanwhile, the power consumption of A^3 is measured moderate and acceptable for long runs on commodity smartphones.

The contribution of this work includes (1) detailed studies to understand the basic performance of smartphone IMU sensors and their sensitivity to environments; (2) based on such understanding, a novel and practical smartphone attitude estimation method which intelligently exploits the sensing redundancy of the gyroscope, compass and accelerometer; and (3) a prototype system of A^3 on Android platform which outperforms existing competitors in various environments and conditions. To our knowledge, A^3 significantly pushes forward the state-of-the-art of phone attitude estimation. A^3 makes it practically feasible to instantly obtain the phone attitude in free motion and for long runs.

The rest of the paper is organized as follows. §2 presents the background and motivation of our problem. §3 describes the principle of MEMS gyroscopes in attitude tracking and characterizes its performance. §4 studies the compass sensor and accelerometer, and describes how they can be leveraged to calibrate gyroscopes. §5 describes the opportunistic calibration technique and presents the final A^3 system design. §6 presents the experimental evaluation results. §7 discusses the related works and §8 concludes this paper.

2. BACKGROUND AND MOTIVATION

Instantly knowing the phone attitude is essential to many applications. For example, the phone attitude can be treated as a key user input for enhanced entertainment in many gesture-based mobile phone games. In 3D photography, the instant phone attitude information helps to retrieve accurate depth information from the pictures taken. In recent dead reckoning based localization and navigation studies, tracking the phone attitude yields the heading of people movement for trajectory mapping.

App	Type	Time	Error
3D camera (Android)	3D photography	4mins	40°
Seene (iPhone)	3D photography	3mins	25°
Sensor Box (Android)	mobile app	3 mins	65°
Jenga (Android)	game	6 mins	35°
Jenga (iPhone)	game	5 mins	30°
Showdown (iPhone)	game	10 mins	38°

Table 1: The attitude estimation error measured from popular apps.

The gyroscopes have been playing a critical role in estimating phone attitude since the first integration to smartphones in 2010[2]. Recent use experience, however, suggests that poor accuracy can be achieved for long period of tracking. Table 1 lists the measured errors of some popular apps from Google Play and Apple Store. The apps typically bear $25^\circ\sim65^\circ$ errors within less than 10 minutes, which significantly impair the use experience. For example, the 3D photography apps like the "3D Camera" for Android and "Seene" for iPhone track the phone attitude when users take multiple pictures and then combine different pictures into one based on the estimated phone attitude. The big attitude estimation error can result in obvious distortion of the final generated picture. In the "Jenga" game, the phone attitude is tracked and taken as input. The user can change the perspective of view to fully inspect a 3D Jenga tower by rotating the phone. The large cumulative tracking errors require frequent reset of the phone attitude from the user. In dead reckoning based localization, accurately detecting the user heading is important but remains unsolved. As the user walks, the localization error due to the heading error is rapidly accumulated, which is up to 20m within 10 mins according to recent studies [22, 24].

One cause of such errors is the limited precision of the state-of-the-art MEMS gyroscope. Compared with high precision gyroscopes like the Fiber Optic Gyroscope (FOG) that can achieve stability down to $10^{-4^\circ}/h$, the MEMS gyroscope is manufactured with low cost based on the Coriolis Vibrating Gyroscope (CVG) principle. In CVG, as the plane of oscillation is rotated, the response detected by the transducer is due to the Coriolis term in its equations of motion. CVG translates the response into the instant 3-axis angular velocities of the phone and continues integration has to be performed on the output to obtain angular increments over the previous attitude. Instantly detecting the absolute attitude thus relies on precise tracking of phone attitude changes, which is known of substantial cumulative errors.

Although no much progress has been made for improving the attitude detection accuracy in the field of mobile phone study, many efforts have been made in robotics domain to address a similar problem, i.e., estimating rigid body orientation. Recent studies in robotics [14, 8, 16, 15] explore the capabilities of magnetometers and accelerometers in orientation estimation, and integrate with the gyroscopes to improve the accuracy. The devised MARG (Magnetic, Angular Rate, and Gravity) orientation for robotic systems performs sensor fusion with input from all three types of sensors to derive the final estimation. Detailed review of related works will be given in Section 7. The performance and characteristics of the sensors used in robotic systems are different from those used on smartphones. For example, compared with the MEMS gyroscope on smartphones, the mechanical and START [1, 8, 21] gyroscopes

[2]iPhone 4 was the first smartphone model that integrates the 3-axis gyroscope. Gyroscopes were later widely integrated in other Android and Windows Phone devices.

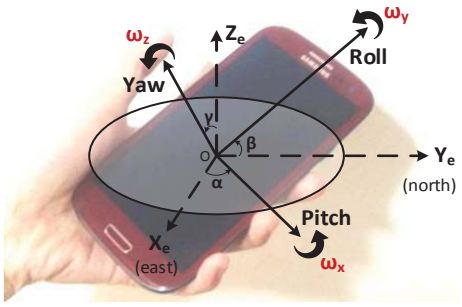


Figure 2: The output of the MEMS gyroscope is 3 angular velocities around the $Roll$, Yaw, and $Pitch$ axis in the phone body-frame.

used in robotic systems are based on different physical rationales, and usually have higher accuracy but different frequency responses to motions. No existing studies, however, have quantitatively understood the nature of smartphone MEMS sensors and their performance in different condition ranges. As a result, straightforward sensor fusion, e.g., Kalman-based approaches [14, 8, 16] with unsorted sensor input, cannot guarantee high accuracy. Besides, the motion patterns and usage environments of robotic systems are also very different with those of smartphones. To our knowledge, simply migrating the approaches for robotic systems to smartphone platforms results in suboptimal performance. For example, the Android API `getRotationMatrixFromVector()` directly adopts existing Kalman-based sensor fusion algorithms to calculate the phone attitude. As the experimental evaluation reveals, the estimation error of Android API may reach 40° or higher in many cases.

In this paper, we comprehensively characterize the nature of the MEMS gyroscopes on modern smartphones, and derive optimized settings and techniques to improve the gyroscope performance in practical working conditions. We then incorporate the compass and accelerometer. While the gyroscope detects 3-axis angular velocities, being able to remove 3 degrees of freedom, the compass measures the geomagnetic north and the 3-axis accelerometer can derive the gravity direction from the 3 axis linear acceleration, providing the capability of removing another 3 degrees of freedom. We thus develop an intelligent calibration approach which selects the best 3 degrees of freedom to determine the phone attitude based on quantitative error estimation of the three different IMU sensors under different condition ranges. Our approach roots in the comprehensive understanding of smartphone IMU sensors and thus fundamentally outperforms existing competing solutions.

3. UNDERSTANDING MEMS GYROSCOPE

The MEMS gyroscope used in smartphones detects the 3-axis angular velocities of the phone. To derive the instant phone attitude, we need to perform continuous integration on the angular velocities.

3.1 Angular Velocity Integration

We deal with two coordinate systems in deriving the phone attitude. One is the earth coordinate system (we call "geo-frame" in this paper) and the other is the smartphone body coordinate system (we call "body-frame" in this paper). The goal is to get the phone attitude in the geo-frame, i.e., to calculate the relative difference between the body-frame and geo-frame. Figure 2 depicts the output of the MEMS gyroscope, which is real-time angular velocities (ω_x, ω_y, and ω_z) around the $Roll$, Yaw, and $Pitch$ axis in the smartphone body-frame. X_e (pointing to the Earth east), Y_e (pointing to the earth north) and Z_e (parallel with the gravity) are the three reference axes in the geo-frame. With continuous integration, the phone attitude can be calculated and represented as the relative difference of the two coordinate systems, which can be described by a rotation matrix, or the angles between the three supporting axes in the two frames, i.e., α, β, and γ (shown in Figure 2).

The MEMS gyroscope has been widely blamed for its poor performance and rapid error cumulation. In this study, however, we find many existing works did not make full efforts to comprehensively understand the MEMS gyroscope working rationale and performance characteristics, leading to suboptimal or inappropriate use of them. For example, the following presents a segment of the code for integrating the 3-axis angular velocities provided in "Android Developers" site [2].

```
//Axis of the rotation sample
float axisX = event.values[0];
float axisY = event.values[1];
float axisZ = event.values[2];
// Calculate the angular speed of the sample
float omegaMagnitude = sqrt(axisX*axisX +
    axisY*axisY + axisZ*axisZ);
// Normalization
...
// Integrate around this axis with the angular
    speed by the timestep into a quaternion
float thetaOverTwo = omegaMagnitude * dT / 2.0f;
float sinThetaOverTwo = sin(thetaOverTwo);
float cosThetaOverTwo = cos(thetaOverTwo);
deltaRotationVector[0] = sinThetaOverTwo * axisX;
...
```

According to Euler's rotation theorem, any rigid body rotation can be represented by a single rotation about some rotation axes. The above code assumes a fixed rotation axis[3] in the body-frame at any particular integration interval, and calculate `omegaMagnitude` as the resultant rotation velocity of the phone. In practice, as the phone rotates, the rotation axis of the phone keeps changing in the body-frame and estimating rotation axis in the geo-frame using the resultant velocity in the body-frame is inappropriate. Such a method results in significant error accumulation, especially when the sampling rate cannot keep up with the shift of phone rotation (which unfortunately is true in most cases as the sampling rate is usually capped at around 600Hz for the MEMS gyroscope). Above code logics have been practiced in the current Android OS, i.e., `handleGyro()` [3] in operating the gyroscope output to derive the attitude for its API. The method calculates the rotation axis in the geo-frame based on the phone angular velocity measured in the body-frame.

In this paper we apply the Euler Axis/Angle method to do the integration and tackle the problem from the perspective of the geo-frame, which is fixed during the phone motion. The method finds an equation for the rotation speed in the geo-frame based on differential. As the equation is directly constructed in the geo-frame, there is no need to make assumption on fixed rotation axis. During the integration, the total phone motion time is divided into multiple time slots and the phone rotation is a sequential combination of the rotation within each slot. The mathematical illustration of the method is detailed in Appendix §A.

[3]The rotation of a three-dimensional object is always around an imaginary line which is called the rotation axis which may keep changing during the rotation. The rotation axis may not pass through the object's body.

3.2 Practical Settings and Techniques

Sensor specifications and configuration. The ADIS1626x series of MEMS gyroscopes are widely used in many smartphone models like HTC Sensation series smartphones. Some other gyroscope models like the AGD1 2022 FP6AQ MEMS gyroscope (used in iPhone) has similar performance. In this paper, we primarily study the ADIS1626x series and the performance of other models can be inferred. For a typical MEMS gyroscope, there are multiple sensor dynamic range r selections (e.g., $\pm 320^\circ$/sec, $\pm 160^\circ$/sec and $\pm 80^\circ$/sec for ADIS1626x series) and sensor bandwidth b selections (e.g., 50Hz and 330Hz for ADIS1626x series). The sensor bandwidth sets a cut-off frequency in responding to phone motions. A higher bandwidth setting can measure higher frequency motion and vibration. Table 2 depicts a part of the data sheet of the ADIS1626x series gyroscope. For different selections of the sensor dynamic range and bandwidth, the output noise is different. As shown in Table 2, the output noise under the setting "$r = \pm 80^\circ$/sec, $b = 50$Hz" is 0.1°/sec. The noise level increases when larger dynamic range or higher sensor bandwidth is selected. Due to the high degree of phone motion, "$r = \pm 320^\circ$/sec, $b = 330$Hz" is the default setting for most smartphones. According to the data sheet, the temperature and linear acceleration affect the output noise of the gyroscope as well, which we will later investigate for practical understanding.

Practical settings and techniques. According to the default setting in smartphones ($r = \pm 320^\circ$/sec and $b = 330$Hz) and the Nyquist theorem, we set the sensor sampling rate to be 660Hz. Typically the MEMS gyroscope cannot support higher sampling rate due to hardware limitations. Determining the window size T_w for integrating the angular velocities needs careful consideration. Generally small T_w provides finer granularity in performing the integration but too small T_w leads to aggressive computation which is not necessary if the phone motion is low and may exceed the processing capacity of the phone. On the other hand, as long as T_w is set fixed, no matter how small it is there are always chances that the angular velocities suddenly change within the window, leading to inaccurate integration. In this paper, we apply adaptive integration interval for performing angular velocity integration. We use the Euler Axis/Angle method introduced in §3.1 to integrate the measured angular velocity for each individual interval when the angular velocities do not change (or the change is sufficiently small). Typically, the interval width is about 5ms$\sim$50ms, which depends on the specific phone motion. In our implementation with Android smartphones, the callback function `onSensorChanged()` in `SensorEventListener` is used to determine whether the angular velocities change and thus the current integration interval. The angular velocity samples are continuously monitored and reported only when the sensor readings change.

Performance. We experiment with the phone model Samsung Galaxy S2 i9100 to examine the proposed method and techniques in phone attitude estimation. In the experiment, we initialize the phone with a preset attitude. We hold the phone in hands and take 1 minute and 5 minute walks in the lab, respectively, after which we put the phone back to its original position and attitude, and measure the estimation error. We take 20 runs for each test, and compare the errors of our method and the method with the Android implementation shown in §3.1. The attitude estimation error is a rotation matrix describing the difference between the ground truth and the estimated result. To visualize the error, we plot the biggest angle error from the 3 axes. This gauge is used throughout the paper to describe the attitude estimation error. Figure 3 compares the errors of both methods. For the 1 minute walks, the 90th percentile and medium errors of our method are 7° and 3°, respectively. For the Android method, the 90th percentile and medium errors, however, are much higher, 27.5° and 19.8°, respectively. For the 5 minute walks, the 90th percentile and medium errors of our method are 31.2° and 22.3°, respectively, and those of the Android method are up to 103° and 79°, respectively. From the experiment results, we see that the MEMS gyroscope is able to provide significantly improved quality of attitude tracking with the Euler Axis/Angle method and our optimization techniques.

Parameter	Sensor Settings/Conditions	Typ	Unit
Output noise	$r = \pm 320^\circ$/sec, $b = 330$Hz	0.9	$^\circ$/sec (rms)
	$r = \pm 320^\circ$/sec, $b = 50$Hz	0.4	$^\circ$/sec (rms)
	$r = \pm 160^\circ$/sec, $b = 50$Hz	0.2	$^\circ$/sec (rms)
	$r = \pm 80^\circ$/sec, $b = 50$Hz	0.1	$^\circ$/sec (rms)
Temperature coefficient	ADIS 1626x	0.005	$^\circ$/sec/$^\circ$C
Linear acceleration	Any axis	0.2	$^\circ$/sec/g

Table 2: The key specifications of the ADIS1626x series of MEMS gyroscopes. The "Typ" column gives the typical output noise for a certain setting. r is the sensor dynamic range and b is the sensor bandwidth (also called "cut-off frequency").

3.3 Performance Characterization

We further do controlled experiments to understand the MEMS gyroscope performance in different condition ranges of different environmental factors.

Temperature. The temperature for the best MEMS gyroscope performance is 25°C. For MEMS gyroscopes in smartphones, however, the temperature compensation has already been done in the sensor chip with an embedded temperature sensor. The calibration temperature range is -40°C$\sim$+85°C with a single temperature point calibration. As suggested in Table 2, the temperature coefficient is merely 0.005°/sec/°C. In order to better understand the impact of temperature, we vary the environment temperature from -20°C to +45°C and study the sensor performance. According to the experiment result, the estimation error caused by the temperature variation lies within a very small range ($<0.002^\circ$/sec/°C) and is negligible. Thus different from other possible working scenarios, the temperature is not a main influencing factor for MEMS gyroscopes in smartphone applications.

Time. According to the rationale of MEMS gyroscopes in deriving the phone attitude, the tracking error accumulates when angular velocity integration is performed. We perform a set of experiments to understand how the time impacts on the error accumulation. We initialize the phone in a preset attitude, and then use the phone in random ways (e.g., playing phone games, walking with the phone in the pocket). After time t, we put the phone back to the original attitude. We repeat the experiment 20$\sim$30 times for $t = 10$s, 1 minute, 5 minutes and 10 minutes, respectively. Figure 4 plots the statistical results about the tracking errors. We see that the tracking error for 10s period is very small. The 90th percentile error is only 3.8°. The tracking error grows up as the time increases. When $t = 1$ minute, the 90th percentile and medium errors are 13.9° and 5.9°, respectively. When $t = 5$ minutes, the 90th percentile and medium tracking errors grow up to 38.7° and 29.4°, respectively. After 10 minutes, the medium tracking error becomes 39.2° and the minimum tracking error is bigger than 21° that is unacceptable in practical usage. The experiment results demonstrate that the attitude tracking error is generally accurate for short time (e.g., less than 1 minute) but may accumulate substantially with time.

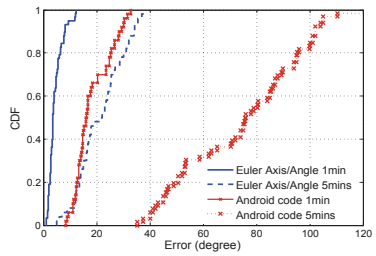


Figure 3: Phone attitude estimation results of different methods.

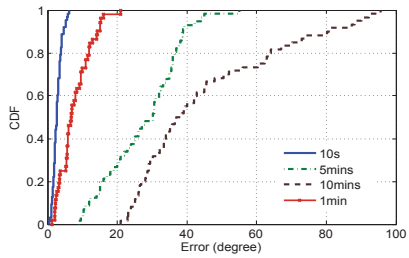


Figure 4: MEMS gyroscope error for different tracking time.

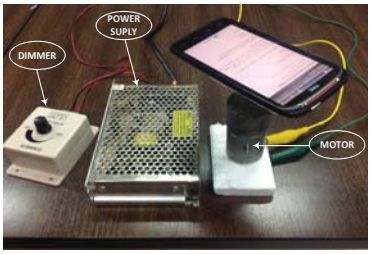
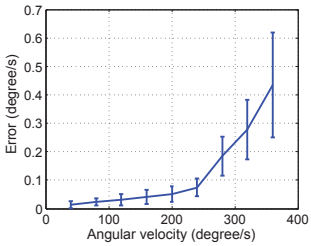
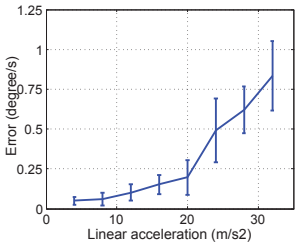


Figure 5: Gyroscope measurement devices.



(a) Error v.s. angular velocity.



(b) Error v.s. linear acceleration.

Figure 6: Impact of (a) angular velocity and (b) linear acceleration on the MEMS gyroscope performance.

Motion. The smartphone motion is typically a combination of rotational and translational motion. Thus we investigate two motion factors, i.e., the angular velocity ω and linear acceleration a. Although the translational motion does not change the phone attitude, it may influence gyroscope performance. Table 2 suggests that the sensor output error caused by linear acceleration is 0.2°/sec/g, but the linear acceleration range, within which the gyroscope can work comfortably, is unclear. Meanwhile, what is the impact of the angular velocity within and out of the dynamic range ($\pm 320^\circ$/sec) needs further study.

To investigate the impact of angular velocity, we put the phone on a rotation plate, the rotation velocity of which is controlled by an adjustable motor (shown in Figure 5). The motor is controlled through a sliding rheostat by adjusting the input voltage. We test different velocities and for each velocity setting, we rotate the phone 30 rounds and repeat the experiment for 10 times. To investigate the impact of linear acceleration, we put the phone in a sloping track and force it to speed. We intentionally control the slope and thrust force to provide different linear accelerations. The actual runtime acceleration is measured by the accelerometer.

Figure 6(a) and (b) plot the tracking error under different settings, which is the accumulated error in degree per second. Figure 6(a) shows that although the dynamic range of angular velocity is $\pm 320^\circ$/sec, when $\omega < 240^\circ$/sec, the error cumulation is small, stable, and roughly proportional to the angular velocity, usually smaller than 0.08°/sec. When $\omega > 240^\circ$/sec, the rate jumps up significantly. It could be as big as 0.2°/sec for $\omega = 280^\circ$/sec and 0.4°/sec for $\omega = 360^\circ$/sec, respectively. Meanwhile, the error exhibits significant variation and becomes hard to predict. Figure 6(b) shows the error cumulation caused by the linear acceleration is relatively higher. When $a < 2g$, the error cumulation rate is relatively stable, roughly proportional to the instant acceleration and usually smaller than 0.2°/sec. When $a > 2g$, the output error quickly rises and becomes uncontrolled. It can be as high as 0.5°/sec for $a = 2.4g$ and 0.75°/sec for $a = 3.2g$, respectively.

We see that for both motion factors, the gyroscope tracking error is relatively small, stable and controlled for a certain safe range, e.g., $\omega < 240^\circ$/sec and $a < 2g$. When those "out-of-range motions" occur, the error rapidly goes up and becomes unbounded to predict. Such out-of-range smartphone motions are common in daily smartphone usage such as strong swings when playing mobile games, vibration and rotation in the pocket during running, etc. It significantly pollutes the consequent attitude estimation result as the phone attitude is calculated with continuous integration.

Based on the experiments, the MEMS gyroscope performance can be summarized as follows:

- The attitude tracking error accumulates as the time increases. If the phone motion is within the safe range and sensor bandwidth, the gyroscope tracking result is accurate in a short time period.

- The attitude tracking error is highly related to the phone motions, i.e., the angular velocity and linear acceleration. The high out-of-range motion significantly pollutes the consequent attitude tracking result.

4. INSTANT ATTITUDE DETECTION

The basic MEMS gyroscope performance cannot provide continuous high accuracy attitude detection due to the error cumulation and out-of-range phone motions. In this section, we describe how we incorporate the independent measurements from the compass and accelerometer to assist the gyroscope in tracking the phone attitude.

4.1 Compass and Accelerometer

Compass. The compass measures the geomagnetic north from the detected geo-magnetic field. The accuracy, however, is unstable, especially indoors where steel structures and electrical appliances may significantly distort the geo-magnetic field. Recent studies report reasonably high compass accuracy for outdoor usage but complicated performance for indoor usage [11, 27]. Figure 7 plots the estimated earth north using the compass indoors and outdoors. In the experiment, we keep the phone attitude unchanged and move around indoors and outdoors. The compass earth north estimation is recorded and then compared to the ground truth to calculate the error. For most of the time, we see accurate outdoor compass output, i.e., $< 5^\circ$ error. The indoor compass output is much more complicated that it could be more than 50° off the ground truth (from the 30th to 60th second) or as low as a few degree error (from the start to 10th second) at particular moments. Generally, the outdoor compass output is of reasonable quality and much more reliable than the indoor output. The indoor output accuracy varies significantly depending on the instant ambient environment.

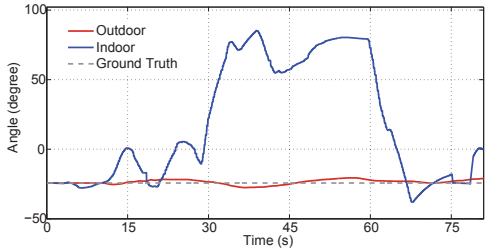


Figure 7: Indoor and outdoor compass performance.

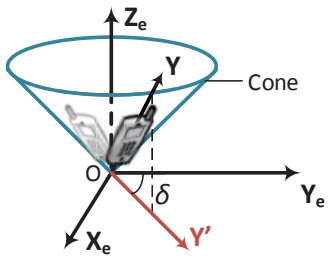


Figure 8: Attitude from the compass and gravity.

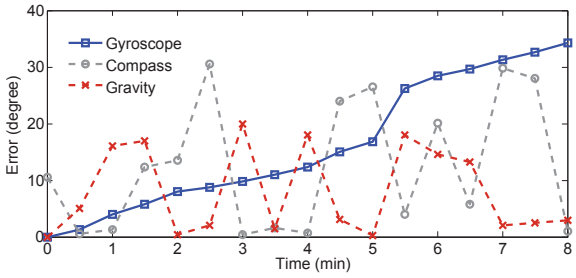


Figure 9: The performance of gyroscope, compass and accelerometer in an instant trajectory.

Accelerometer. Most smartphone platforms provide the API of retrieving gravity (e.g., `Sensor.TYPE_GRAVITY` in Android and `GLGravity` in Apple iOS) from the accelerometer readings. With the help of the powerful Butterworth filter [9], the acceleration caused by phone motion can be filtered out and the frequency components of gravity can be extracted, i.e., the direction of Y_e in the smartphone body-frame. We denote the components of gravity along the X, Y and Z axis to be (g_x, g_y, g_z), where $\sqrt{g_x^2 + g_y^2 + g_z^2} = g$. The filter cut-off frequency is normalized to 1 radian/sec and the frequency response (gain) is

$$G(\omega_0) = \sqrt{\frac{1}{1 + {\omega_0}^{2n}}}, \quad (1)$$

where ω_0 is the angular frequency and n is the number of poles in the filter. The smartphone motion is typically composed of the rotational and translational parts, both of which may cause acceleration variation along the 3 axes. According to the Butterworth filter rationale, if the noise frequency is higher than its cut-off frequency, the high frequency signal caused by the translational motion can be filtered out. The noise caused by the phone rotation, however, cannot be fully filtered out if the rotation frequency is high. The frequency response (gain) $G(\omega_0)$ of Butterworth filter becomes small when ω_0 gets big. Generally the gravity can be accurately extracted when the phone rotation is slight and no constant linear acceleration is imposed. A specific example is when the phone is put static.

Attitude from compass and gravity. As depicted in Figure 8, given the direction of gravity on the phone body-frame, the phone attitude is constrained on a conical surface in the geo-frame. On the other hand, the compass outputs the angle δ between Y' and Y_e axis (pointing to the earth north) in the geo-frame, where Y' is the projection of Y axis of the body-frame on the X_e-Y_e plane of the geo-frame. Considering the angle δ between Y' and Y_e, we can thus uniquely fix the phone attitude on the conical surface. This provides us an alternative of removing 3 degrees of freedom to determine the phone attitude if we have accurate compass and gravity output. The detailed construction of the rotation matrix for the phone attitude is provided in Appendix §B.

Compared with gyroscope. The attitude estimation from the compass and accelerometer is independent and of different nature compared with the result from the gyroscope. The compass and accelerometer give instant status estimation which is unrelated to any previous estimations, while the gyroscope gives a cumulative estimation of the attitude through continuous integration on angular velocities. Figure 9 compares the attitude estimation from the gyroscope and that from the compass and accelerometer during an 8 minute walk. We see a clear difference in the natures of their estimations. In most of the time, the gyroscope produces small estimation errors which, however, accumulate with time. A few sudden jumps of the error (e.g., in the 5th minute, probably due to out-of-range phone motion) significantly contribute to the final cumulative error. On the other hand, the compass and accelerometer perform in very different ways. They have the opportunity to produce a few very good estimations but many bad ones as well. The estimation errors depend on instant phone statuses and do not accumulate. Such different natures in their performance provide us the opportunity to complement the gyroscope estimation with calibration from the compass and accelerometer at "good" moments.

4.2 Putting It All Together

We incorporate the three IMU sensors and propose A^3, an accurate and automatic attitude detector to continuously estimate the phone attitude. Figure 10 illustrates the A^3 system architecture. There are two major components: gyroscope tracking and calibration. As depicted in Figure 10 (left), A^3 tracks the phone attitude using the gyroscope. The angular velocities are adaptively integrated to calculate the phone attitude (the rotation matrix $\mathbf{R}$). The tracking error E_g of the gyroscope is carefully estimated based on the real time monitoring of the phone motion. As summarized in §3.3, the MEMS gyroscope error is mostly related to the two motion parameters, i.e., the angular velocity ω and linear acceleration a. In this paper, we assume the impact from the two parameters is independent, i.e., the phone's rotational and translational motion independently affects the measurement of Coriolis vibration. For each integration interval i, the accumulated error e_i is thus measured as

$$e_i = f_\omega(\omega_i)\Delta t_i + f_a(a_i)\Delta t_i,$$

where f_ω is a functional relationship between the angular velocity ω_i and the error, and f_a is a functional relationship between the linear acceleration a_i and the error. Δt_i is the length of the tracking interval. The gyroscope tracking error at time t_x is accumulated as

$$E_g(t_x) = \sum_{t=t_0}^{t_x} e_i = \sum_{t=t_0}^{t_x} [f_\omega(\omega_i)\Delta t_i + f_a(a_i)\Delta t_i], \quad (2)$$

where t_0 is the start point of current tracking. f_ω and f_a can be determined according to the experimental understanding from Figure 6. When the angular velocity $\omega < 240°$/sec, the estimation error follows a Gaussian distribution with limited variation. The

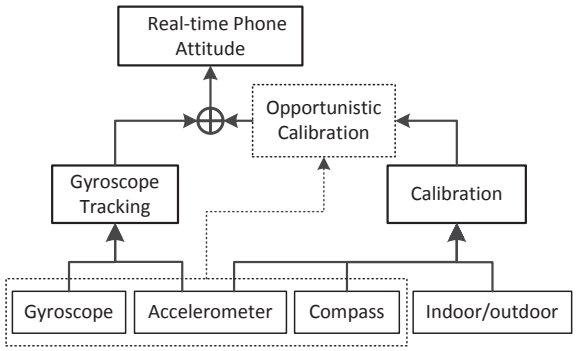


Figure 10: A^3 architecture.

mean linearly increases as ω increases towards 240°/sec. When $\omega > 240°$/sec, the error rises much higher with intensified variation, being hard to predict. Thus in A^3 system, we set f_ω linear for the safe range when $\omega < 240°$/sec and unbounded when $\omega > 240°$/sec:

$$f_\omega(\omega) = \begin{cases} 0.0003\omega, & \text{if } \omega < 240°/\text{sec} \\ unbounded, & \text{if } \omega \geq 240°/\text{sec}. \end{cases}$$

The impact of the linear acceleration a is similar with the angular velocity , with $a < 2g$ a safe range where the error linearly increases with a and $a > 2g$ of unbounded error:

$$f_a(a) = \begin{cases} 0.001a, & \text{if } a < 2g \\ unbounded, & \text{if } a \geq 2g. \end{cases}$$

Above error estimation is based our experimental understanding of ADIS1626x series gyroscope which is the most widely adopted MEMS gyroscope. Other gyroscopes can be parameterized slightly differently, but in the same essence when experimental results are available. As depicted in Figure 10 (right), A^3 automatically detects good calibration opportunities when the compass output and the extracted gravity direction are accurate.

According to our understanding on the compass rationale, the indoor/outdoor information can be effectively used to indicate the compass accuracy. We use a simplified version of IODetector [27] to perform indoor/outdoor detection with the light sensor and cellular module on smartphones. The compass output is considered accurate only when strict outdoor context is detected. For the gravity output to be accurate, we set a rigorous condition on the instant angular velocity, i.e., $\omega < 15°$/sec. When both the compass and gravity outputs are obtained in valid conditions, we confirm a calibration opportunity. The phone attitude derived from the compass and gravity calibrates what obtained from gyroscope tracking if its estimated tracking error $E_g > 5°$. The consequent gyroscope tracking is then carried on the calibrated attitude basis to produce real time phone attitude.

5. OPPORTUNISTIC CALIBRATION

The proposed calibration method sets rigorous conditions in qualifying the calibration opportunities so the accuracy is guaranteed. In practice, the rigorous conditions result in too few opportunities to timely calibrate gyroscope drifts, e.g., no calibration indoors or when the phone motion is intense. Those cases, however, are common for most attitude based applications, e.g., playing games with intense phone motion, indoor localization, etc.

Simply lowering the condition requirements, on the other hand, may harm the calibration quality and thus the accuracy of attitude estimation. In this section, we introduce an opportunistic calibration technique to identify more "good" opportunities. The basic idea is that we leverage the gyroscope estimation to capture phone attitude dynamics. As our experiments reveal in §3.3, the gyroscope can provide very accurate attitude tracking in short time periods (e.g., within 10s). Although the instant attitude estimation of gyroscope may not be accurate due to base errors from previous states, the estimated attitude change is accurate for most of the time (when phone motion is within the safe range of the gyroscope), which sets a very good reference. We compare with the attitude change derived from the compass and gravity, and if both estimations derive the same change of phone attitude we believe the compass and gravity make an accurate attitude estimation. As the output of the compass and gravity is the instant attitude estimation, which is independent of previous states of the phone, we can then use it to reset the current attitude estimation and continue the gyroscope estimation from the new attitude base.

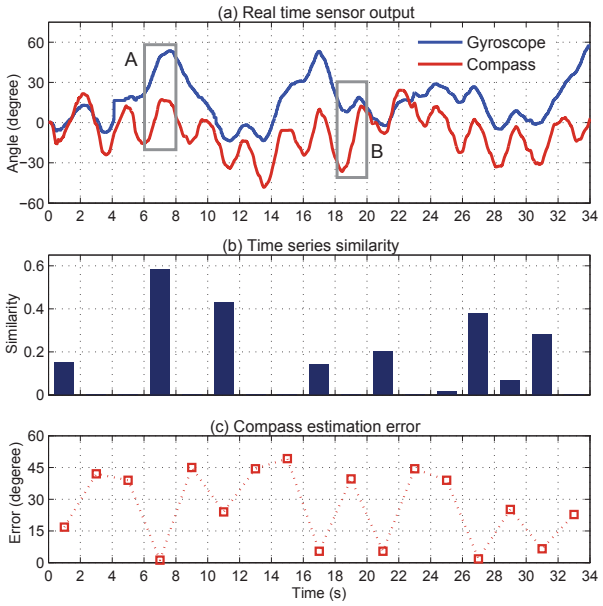


Figure 11: The time series similarity of compass and gyroscope sensor output, and the related compass error.

We denote C as the compass output of the earth north, G as the gravity direction extracted from the accelerometer, and S as the attitude estimation from the gyroscope. We compare the time series similarity of the changes of C and S as well as G and S in the period. If the changes of C and G are "parallel" with S, we can validate their instant accuracy. We examine how C and G are "parallel" instead of "correlated" with S to validate their quality because it indicates a stronger degree of synchronization between two time series signals. For each detection window d, suppose the compass output is $C = \{c_1, c_2, \cdots, c_n\}$ and the corresponding earth north direction extracted from the gyroscope estimation is $S_c = \{s_{c1}, s_{c2}, \cdots, s_{cn}\}$. We calculate their time series similarity p as

$$p = \frac{1}{2^{Var(S_c - C)}}, \quad (3)$$

where $Var(S_c - C)$ is the variance of their difference. $p = 1$ indicates the highest parallel degree. A bigger p indicates a higher similarity between the two.

Figure 11 depicts the results of the gyroscope and compass output on earth north for a half minute walk in an indoor office. We set the similarity detection window of 2 second width. Due to the exponential property of Equation (3), we see that for most of the time

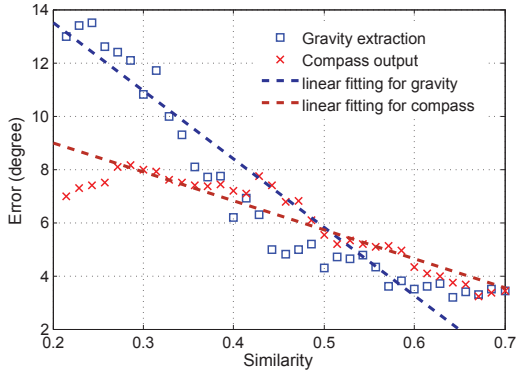


Figure 12: The calibration error for different similarities.

the similarity $p \approx 0$ and only when the compass output changes in a very similar trend with the gyroscope output, $p > 0.2$. For example, in the detection window A in Figure 11(a) and (b), the compass output is almost parallel to the gyroscope output with $p = 0.6$. On the contrary, in the detection window B, although the two outputs are highly correlated with each other, their time series similarity is smaller than 0.02 and not parallel.

Figure 11(c) summarizes the compass output error measured in the experiment trace. We see that a big p generally corresponds to a small output error of the compass (e.g., the 6th∼8th second and 26th∼28th second). This may not always be true, e.g., at the window of 10th∼12th second, the compass output error reaches almost 29° although the similarity p is as high as 0.5, because the phone motion at that moment exceeds the safe range of the gyroscope. In order to guarantee the quality of the calibration, we set the requirement of $\omega < 240°$/sec and $a < 2g$. so the gyroscope estimation is guaranteed a truthful reference. We look after the gravity estimation from the accelerometer in a similar procedure. We test the similarity of G and S with Equation (3) but project to the gravity direction. We denote p_c as the similarity of C and S, and p_g as the similarity of G and S for each detection window of 2 seconds. We confirm the detection of a calibration opportunity if both p_c and $p_g > 0.2$.

For the valid calibration opportunities, we estimate the error of the instant phone attitude derived from the compass and accelerometer. We primarily look at the values of p_c and p_g, as they indicate how well the instant attitude estimation conforms to the truthful reference. In order to figure out the error of each calibration opportunity according to the similarity, we experiment with 3 types of smartphones across various conditions to learn their relationship. We experiment with various phone usage patterns such as walking, running, and playing phone games, etc. We collect the sensor data in 21 indoor and 9 outdoor sites, respectively. For each site, we repeat the experiment around 10 times. Figure 12 plots our statistical average of measurement results. For both the gravity and compass output, the error decreases as expected when the similarity increases. Linear fittings well approximate such relationships. We see that the gravity extraction error is bigger than the compass error when similarity $p < 0.5$ and smaller when $p > 0.5$. The error of their combination is influenced by both compass estimation and gravity extraction. Thus we estimate the error of compass E_1 and error of gravity E_2 through linear fitting separately.

$$E_1 = -32.14p_c + 19.93,\ E_2 = -12.86p_g + 11.57.$$

The error of their combination E_c is estimated as $E_c = \max\{E_1, E_2\}$, where E_c fits to the bigger error of the two. An opportunistic calibration is qualified if $E_c < E_g$. Algorithm 1 presents the high-level pseudo code of A3 opportunistic calibration algorithm.

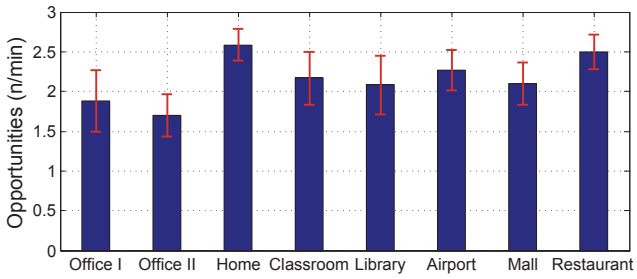


Figure 13: The number of calibration opportunities detected at different places.

Algorithm 1 A^3 Opportunistic Calibration Algorithm

Input:
Rotation matrix R_g from the gyroscope and R_c from the combination of gravity and compass;
Estimation error of the gyroscope E_g and error of the combination of gravity and compass E_c;
Similarity parameter p_c and p_g;
Output:
Final rotation matrix R.

```
if p_c > 0.2 and p_g > 0.2 then
    if E_c < E_g then
        R ← R_c
        R_g ← R_c
    else
        R ← R_g
else
    R ← R_g
```

Opportunistic calibration provides a much broader range of calibration opportunities, not necessarily constrained in modest usage and outdoors. We do extensive experiments to examine different indoor places when walking with the phone free in the pocket. Figure 13 plots the number of qualified calibration opportunities detected per minute. There are in average 2 opportunities detected per minute with the highest at home (2.6) and the lowest in office II (1.7). Such opportunities are abundant for timely gyroscope calibration, considering the less than 5° attitude tracking error per minute.

The final A^3 system employs the opportunistic calibration (dashed part in Figure 10) which overrides the opportunity detection method in §4.2.

6. EVALUATION

We implement A^3 on the Android platform and experiment with three different phone models and under different conditions. We first present the experiment devices and settings in §6.1. We demonstrate detailed system performance of a typical experiment trial and report statistical performance for different scenarios in §6.2. We study heading estimation as a particular application of A^3 and compare the performance with existing techniques in §6.3. We compare A^3 performance with several popular smartphone apps and games in §6.4. We investigate the power consumption of A^3 in §6.5. The following details the experiment methodology and the evaluation results.

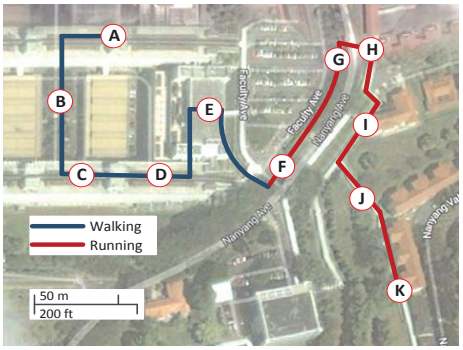


Figure 14: An instant experimental trace in our campus.

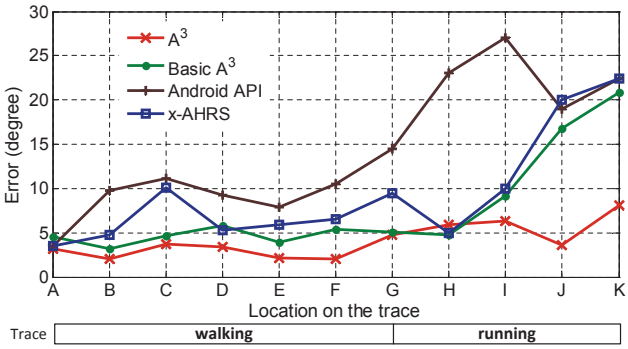


Figure 15: Attitude estimation error of different methods along the experimental trace.

6.1 Experiment Devices and Settings

Smartphones. We implement A^3 on Android 4.2 platform and experiment with three different types of smartphones including HTC Sensation XE, Samsung Galaxy S2 i9100, and LG Google Nexus 4, all of which are equipped with MEMS gyroscopes, accelerometers and magnetism sensors, etc. Their RAM and CPU capacity can easily support the computation of A^3. As A^3 is independent of platforms, we believe it can be easily implanted to other OS platforms or phone models such as Windows Phone and Apple iOS based smartphones.

Experiment settings. We primarily evaluate the attitude estimation performance in three different scenarios, including walking with the phone in hand, walking with the phone free in the pocket and running with the phone in hand. We experiment at different sites including the office, home, the airport, the shopping mall, laboratory, etc. The error is evaluated as the biggest angle error among the three axes, the same as in previous sections.

Comparison. We conduct comparative experiment to investigate the performance of following approaches:

- **A^3**: The complete implementation of A^3 system.
- **Basic A^3**: Basic A^3 implementation as introduced in 4.2 where the opportunistic calibration is not incorporated.
- **Android API**: The Android API `getRotationMatrixFromVector()` practices the Kalman-based orientation estimation algorithms [14, 8, 16] which perform unsorted sensor fusion. As a system API, it has been invoked in many apps for various applications like indoor localization, navigation and human activity recognition. We do not try to optimize its parameters for being not able to locate any available documents on optimizing the Android API.
- **x-AHRS**: The x-AHRS algorithm [15] is one of the latest orientation estimation algorithms which produces AHRS (Attitude Heading Reference Systems) input for robotic systems and wearable systems. It has been integrated and made commercially available in the x-IMU sensor boards [4]. To our knowledge, x-AHRS produces the best reported attitude estimation till now.

6.2 Performance in Different Conditions

We conduct a number of experiment trials with different use conditions. Figure 14 depicts one trace. The total length of the trace is approximately 700 meters. We experiment with different activities during the trial including walking (from A to G) and running (from G to K). We cannot track the continuous estimation error on the trace because obtaining the continuous ground truth of the phone attitude is not possible. Thus for each intermediate segment, we report the system error at individual testing spots (A-K), where the phone attitude ground truth is manually collected. We perform 5 different attitude estimation methods simultaneously in the trial and summarize their tracking errors in Figure 15.

According to the results in Figure 15, A^3 outperforms all other methods during the entire trace except for point H, where "basic A^3" and "x-AHRS" perform slightly better. This could be due to some suboptimal calibration conducted in A^3. Nevertheless, no apparent performance degrade of A^3 is observed during the entire trial. For all the methods, they perform better for walking segments and worse for running segments. This is because the phone motion when we are walking is much smoother and of lower motion frequency than the running scenario. Thus usually the phone motion is within the gyroscope sensor dynamic range. The Android API produces worst results in all of the four methods across the entire trace. Its attitude estimation error on the walking trace is bounded within 15° but significantly jumps up to 27° on the running segments. The estimation results from the "x-AHRS" algorithm are relatively smooth. The estimation error on the entire walking trace is smaller than 10° and smaller than 5° at A, B and H. However, on the running trace, as the phone motion is of high frequency, x-AHRS cannot optimally fuse the sensor outputs and the error is as high as 20°. Its performance is comparable with "basic A^3" at D and H, but worse than "basic A^3" at other locations. The "basic A^3" performs the closest to A^3, outperforming other approaches at most of the time, which demonstrates the performance gain from the careful gyroscope operation and quality calibration with the compass and accelerometer. The performance gap between "basic A^3" and A^3 tells the gain of opportunistic calibration technique.

For statistical comparison we perform experiments in three scenarios, namely walking with the phone in hand, walking with the phone in pocket, and running with the phone in hand. We examine the attitude tracking error in 20 minute usage. We perform about 40 runs for each scenario. All the 4 methods are performed simultaneously during the each run. The statistical results are displayed in Figure 16∼18. Figure 16 presents the CDF of the estimation errors during walking with the phone in hand. The median estimation errors of Android API, "x-AHRS" , "basic A^3" and A^3 are approximately 17.4°, 11°, 9.5° and 4.2°, respectively. Their 90th percentile errors are 37.1°, 25.3°, 17.9° and 8.3°, respectively. The Android API produces the worst result in such a scenario. Figure 17 presents the CDF of the estimation errors during walking with the phone in pockets. The mobile phone has a higher freedom in the pocket and results in degraded performance for all methods. The median estimation errors of the 4 methods are approximately 28.6°, 20.5°, 16.5° and 7°, respectively. The 90th percentile errors are 53.5°, 32.6°, 21.4° and 11.5°, respectively. Figure 18 the CDF

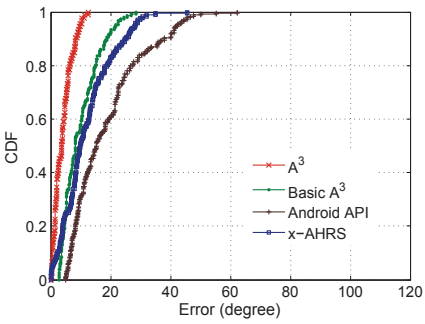


Figure 16: Walking with the phone in hand.

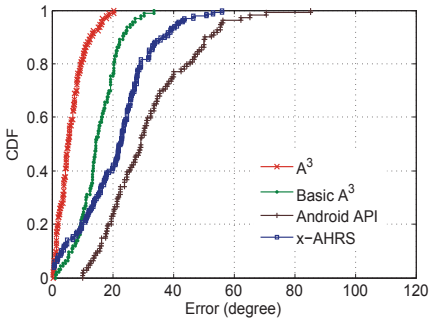


Figure 17: Walking with the phone in the pocket.

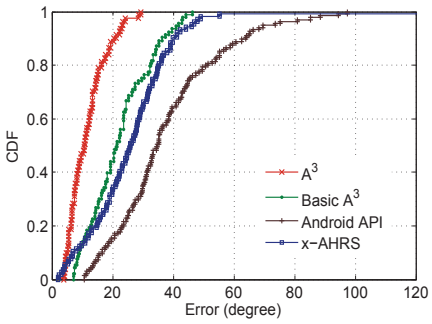


Figure 18: Running with the phone in hand.

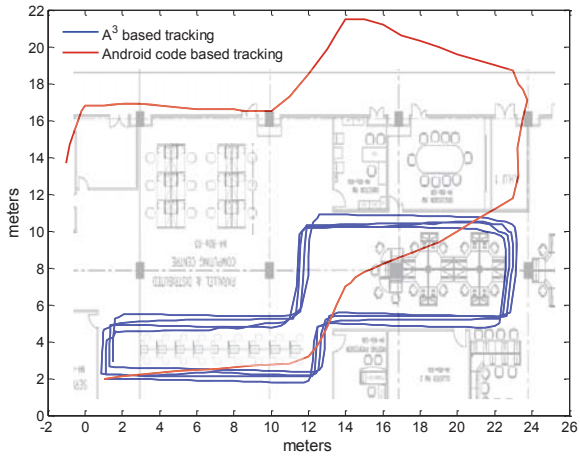


Figure 19: Dead-reckoning based tracking result using A^3 and the Android code shown in §3.1 for heading estimation, respectively.

of the estimation errors during running with the phone in hand. The median estimation errors are approximately 35°, 23.4°, 20.3° and 9.7°, respectively. The 90th percentile errors are 60.6°, 40.5°, 37.4° and 18.5°, respectively. We see that A^3 consistently outperforms the rest in all scenarios. Higher gain can be obtained when more motion freedom of the phone is allowed. Even without opportunistic calibration, the "basic A^3" still slightly outperforms the others in statistics for most of the time. The results demonstrate how the understanding of the IMU sensors and the comprehensive use of them help to provide huge performance gain.

6.3 Application in Heading Estimation

The detected phone attitude can be directly used to estimate the user heading in dead-reckoning. In this section, we conduct a case study to examine how A^3 can be applied to achieve accurate heading estimation. The same as existing works in heading estimation, we assume that the phone is held in hand with the *Roll* axis of the phone pointing to the user heading direction (as a matter of fact, while the phone is held relatively still to the human body we can always transform the phone attitude to extract the heading direction).

We implement a dead-reckoning based tracking approach based on step counting [10, 24], but apply A^3 to estimate the user heading. As Figure 19 depicts, we experiment with such an approach on a 58m circular path in the lab. Heading estimation is very accurate. As a result, after 7 minute walking on the path for 4 rounds, the tracking error is merely 1.3m. We see from Figure 19 that the error is mainly due to the shift from inaccurate distance estimation in steps, not the heading estimation. As a comparison, we also record the result of using the Android code shown in §3.1 for heading estimation. As Figure 19 depicts, the estimated trajectory deviates from the actual path and the error rapidly propagates. After walking for only 1 round, the tracking error accumulates to 13.2m. We see that the error mainly comes from the heading deviation during the turns. We could not directly compare A^3 with some existing dead-reckoning approaches [10, 24] as the source code was not available to us, but according to the publicly reported results, A^3 is highly likely to outperform them.

App	Type	Time	App Error	A^3 Error
Sensor Box	Sensor app	1 min	40°	3.5°
		5 mins	50°	5°
		10 mins	75°	6.4°
Show Down	Game	1 min	9°	5°
		5 mins	30°	7°
		10 mins	35°	6°
Gyroscope Rotate	Sensor app	1 min	10°	5°
		5 mins	28°	8°
		10 mins	45°	4.5°

Table 3: Comparison of A^3 with phone apps/games.

6.4 Comparison with Popular Apps

Many smartphone apps and games detect and take the phone attitude change as input. We also examine the performance of A^3 in comparison with those of several popular apps and games from Google Play, including "Android Sensor Box", "Show Down", and "Gyroscope Rotate". All 3 apps use gyroscope to track the phone attitude dynamics. We test the performance of the three apps in 1 minute, 5 minutes and 10 minutes scale, respectively. For each app, we initialize the phone with a preset attitude and then use it in random ways, after which we put the phone back to the initial attitude. For each app, we let A^3 simultaneously run at the background and compare the attitude tracking accuracy of A^3 with those of the apps. We repeat each experiment for 10 runs and calculate the average estimation error, which is summarized in Table 3.

We see substantially gained performance over all the 3 apps, particularly when the phone is played for a longer time. After 1 minute usage, the attitude estimation error of all the apps vary from 9° to 40°. The error grows up to 28°∼50° after 5 minute usage and 35°∼75° after 10 minute usage. On the contrary, the error of A^3 slight increases with time and maintains within 8°. As we do not have access to the source codes of those apps, we do not have precise knowledge on their approaches in attitude tracking. We specu-

late that none of them make careful efforts in using the gyroscope, nor incorporate any calibration techniques, as their attitude tracking errors apparently accumulate fast. It seems that they did not even invoke the Android API but only used the direct output from the gyroscope.

6.5 Power Consumption

We measure the power consumption of A^3 on Samsung Galaxy S2 i9100 mobile phone. We measure the average working current of A^3 using Monsoon power monitor. The phone's battery capacity is 6.11Wh (1650mAh). We measure the power consumption of the mobile phone under 6 different use cases, i.e., screen off, screen on, sampling sensors only, basic A^3, A^3 and the use of Android API. Figure 20 plots how the working current changes when different operations are performed. The plotted working current is a moving average value of every 0.1 second.

The working current is close to 0 initially when the phone screen is off, and jumps to around 155mA when the screen is on. In order to test the algorithms on the same baseline, we turn off the phone screen from the 100th second and run the algorithms at the background. From the 100th second, the phone starts continuously sampling the sensor readings but does not perform any computation. The sampling rates of different sensors are set to be the same with those of A^3. The average working current is about 80mA. We let the phone run different attitude tracking algorithms from the 130th second. We see comparable overall working current for all three approaches. The basic A^3 algorithm is performed from the 130th to 220th second. The working current increases a little to about 90mA. From the 220th second, A^3 algorithm is running and the average current is about 95mA and that of running Android API is similar. Although different algorithms perform differently in estimating the phone attitude, their power consumption is comparable and slightly higher than "sampling sensors only". According to the result, the data processing of all the algorithms similarly contributes extra amount of around 5∼15mA current. The power consumption of the three algorithms is moderate and acceptable for long runs on commodity smartphones.

7. RELATED WORK

Orientation and attitude estimation. The most related work in the literature is the estimation of the IMU or MARG orientation for wearable systems and robotic systems. Kalman filter was widely used in those works to fuse different sensor inputs. Barshan *et al.* [8] propose to estimate the orientation of the robots for mobile robotics applications using inertial sensor models and an extended Kalman filter. Marins *et al.* [16] present an extended Kalman filter for real-time estimation of rigid body orientation using MARG sensors. Different robotic applications may have different requirements, which ask for specific algorithm and system design. Kong *et al.* [13] present an algorithm for inertial navigation system using a generic error propagation model. Lee *et al.* [14] propose a sensor fusion technique of sensors (gyro and tilt) to measure the balancing angle of the inverted pendulum robot system. Most recently, Madgwick *et al.* [15] develop an orientation estimation algorithm for the x-IMU sensor board [4]. A quaternion representation of orientation is used to describe the coupled nature of orientation in 3 dimensions and fuse the sensor inputs. Those approaches exploit the sensing redundancy of different sensors in attitude detection. Without fundamental understanding of smartphone MEMS sensors and their performance variations with environment dynamics, those approaches cannot make appropriate error control in sensor fusion, and thus cannot provide satisfactory performance when applied to smartphones. Simple parameter optimization on Kalman-based approaches will not address the problem because it is based on the frequency responses of different sensors not the quality of raw sensor readings. The error of raw sensor readings is environment dependent that frequency response cannot fully capture.

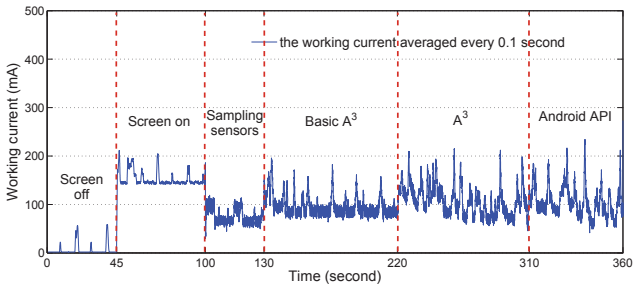


Figure 20: Energy consumption measurement of different algorithms.

Heading estimation. There have been many research works on user heading estimation for indoor localization, navigation and tracking, etc. Walkie-Markie [22] uses the gyroscope to estimate the user heading for indoor pathway mapping. UnLoc [24] makes use of the gyroscope and compass for accurate user heading estimation in dead-reckoning based indoor localization. Headio [23] aggregates the ceiling images of an indoor environment, and by using computer vision-based pattern detection techniques to provide directional references. Afzal *et al.* [7] propose to identify the magnetic field measurements for estimating user heading. As we show in §6.3, the phone attitude output of A^3 can be used to produce the heading estimation with higher accuracy. Besides, A^3 solely utilizes the IMU sensors on the phone and is thus orthogonal to some of above techniques.

Attitude based applications. A^3 benefits a broad range of other applications such as image stabilization [12], 3D photography [18, 5], and phone apps, etc. Karpenko *et al.* [12] perform video stabilization and rolling shutter correction based on real-time attitude estimation using the gyroscope. Oth *et al.* [18] develop a calibration procedure to determine the rolling shutter line delay using the gyroscope. Snapily3D [5] enables 3D camera on smartphone to store images with different aspect ratios, which relies on the gyroscope to estimate the phone attitude. Phone attitude information is vital to a great number of phone apps and games. Significantly improved attitude estimation with A^3 essentially benefit those applications.

8. CONCLUSION

This paper presents A^3 for accurate phone attitude detection. With careful and intelligent use of the gyroscope and other IMU sensors, A^3 makes it possible to estimate the mobile phone attitude in free motion. The experiment results demonstrate A^3 provides far better tracking accuracy than other possible competitors and maintains the high accuracy during long runs. One future work is to explore how well the attitude tracking result can be applied to support other novel applications like localization, tracking, mobile gaming, etc.

9. ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers for their constructive insights and valuable comments for improving the quality of the paper. We acknowledge the support from Singapore MOE AcRF Tier 2 grant MOE2012-T2-1-070 and NTU Nanyang Assistant Professorship (NAP) grant M4080738.020.

10. REFERENCES

[1] Solid State Angular Rate Gyro. http://www.watson-gyro.com/products/rate-gyro-ARS-spec.html.

[2] Andriod Developer Sensor Event. http://developer.android.com/reference/android/hardware/SensorEvent.html.

[3] Android sensor sevice source code. https://android.googlesource.com/platform/frameworks/native/+/master/services/sensorservice/.

[4] x-IMU sensor board. http://www.x-io.co.uk/products/x-imu/.

[5] Snapily 3D. http://www.snapily3d.com/.

[6] Gimbal Lock. http://www.chrobotics.com/library/understanding-quaternions.

[7] M. Afzal, V. Renaudin, and G. Lachapelle. Magnetic field based heading estimation for pedestrian navigation environments. In *Proceedings of IPIN 2011*, pages 1–10, 2011.

[8] B. Barshan and H. Durrant-Whyte. Inertial navigation systems for mobile robots. *IEEE Transactions on Robotics and Automation*, 11(3):328–342, 1995.

[9] S. Butterworth. On the theory of filter amplifiers. *Wireless Engineer*, 7(6):536–541, 1930.

[10] I. Constandache, X. Bao, M. Azizyan, and R. R. Choudhury. Did you see bob?: Human localization using mobile phones. In *Proceedings of ACM MobiCom 2010*, pages 149–160, 2010.

[11] I. Constandache, R. Choudhury, and I. Rhee. Towards mobile phone localization without war-driving. In *Proceedings of IEEE INFOCOM 2010*, pages 1–9, 2010.

[12] A. Karpenko, D. Jacobs, J. Baek, and M. Levoy. Digital video stabilization and rolling shutter correction using gyroscopes. Technical Report CSTR 2011-03, Stanford University, 2011.

[13] X. Kong. Ins algorithm using quaternion model for low cost imu. *Robotics and Autonomous Systems*, 46(4):221 – 246, 2004.

[14] H.-J. Lee and S. Jung. Gyro sensor drift compensation by kalman filter to control a mobile inverted pendulum robot system. In *Proceedings of IEEE ICIT 2009*, pages 1–6, 2009.

[15] S. Madgwick, A. J. L. Harrison, and R. Vaidyanathan. Estimation of imu and marg orientation using a gradient descent algorithm. In *Proceedings of IEEE ICORR 2011*, pages 1–7, 2011.

[16] J. Marins, X. Yun, E. Bachmann, R. Mcghee, and M. Zyda. An extended kalman filter for quaternion-based orientation estimation using marg sensors. In *Proceedings of IEEE/RSJ IROS 2001*, volume 4, pages 2003–2011, 2001.

[17] E. Miluzzo, A. Varshavsky, S. Balakrishnan, and R. R. Choudhury. Tapprints: your finger taps have fingerprints. In *Proceedings of ACM MobiSys 2012*, pages 323–336, 2012.

[18] L. Oth, P. Furgale, L. Kneip, and R. Siegwart. Rolling shutter camera calibration. In *Proceedings of IEEE CVPR*, pages 1360–1367, 2013.

[19] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen. Zee: zero-effort crowdsourcing for indoor localization. In *Proceedings of ACM Mobicom 2012*, pages 293–304, 2012.

[20] M. Shahzad, A. X. Liu, and A. Samuel. Secure unlocking of mobile touch screen devices by simple gestures: You can see it but you can not do it. In *Proceedings of ACM MobiCom 2013*, pages 39–50, 2013.

[21] T. Shelley and J. Barrett. Vibrating gyro to keep cars on route. *Eureka on Campus, Eng. Materials and Design*, 4:17, 1992.

[22] G. Shen, Z. Chen, P. Zhang, T. Moscibroda, and Y. Zhang. Walkie-markie: Indoor pathway mapping made easy. In *Proceedings of USENIX NSDI 2013*, pages 85–98, 2013.

[23] Z. Sun, S. Pan, Y.-C. Su, and P. Zhang. Headio: Zero-configured heading acquisition for indoor mobile devices through multimodal context sensing. In *Proceedings of ACM UbiComp 2013*, pages 33–42, 2013.

[24] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury. No need to war-drive: unsupervised indoor localization. In *Proceedings of ACM MobiSys 2012*, pages 197–210, 2012.

[25] Z. Yang, C. Wu, and Y. Liu. Locating in fingerprint space: Wireless indoor localization with little human intervention. In *Proceedings of ACM Mobicom 2012*, pages 269–280, 2012.

[26] Y. Zheng, G. Shen, L. Li, C. Zhao, M. Li, and F. Zhao. Travi-navi: Self-deployable indoor navigation system. In *Proceedings of ACM MobiCom 2014*, 2014.

[27] P. Zhou, Y. Zheng, Z. Li, M. Li, and G. Shen. Iodetector: a generic service for indoor outdoor detection. In *Proceedings of ACM SenSys 2012*, pages 361–362, 2012.

APPENDIX

A. EULER AXIS/ANGLE METHOD

We use a rotation matrix $\mathbf{R}$ given in terms of a unit vector $\mathbf{e}$ along the rotation axis (called "Euler axis" in Euler's rotation theorem), and the angle θ to describe the rotation of the phone. According to the Rodrigues' rotation formula,

$$\mathbf{R} = \mathbf{I}cos\theta + \mathbf{e}(\mathbf{e}\cdot\mathbf{I})(1-cos\theta) + (\mathbf{e}\times\mathbf{I})sin\theta, \quad (4)$$

where $\mathbf{I}$ is $\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}$ in the geo-frame. The body-frame rotates about $\mathbf{e}$ at the speed of $\frac{d\mathbf{R}(t)}{dt} = \dot{\theta}\mathbf{e}$ in the geo-frame. The rotation speed also can be represented as $\mathbf{R}(t)\Omega(t)$, where $\Omega(t) = \{\omega_x, \omega_y, \omega_z\}$ and $\mathbf{R}(t)$ is the time-varying rotation matrix. Thus we have

$$\frac{d\mathbf{R}(t)}{dt} = \mathbf{R}(t)\Omega(t). \quad (5)$$

There are two possible solutions to calculate $\mathbf{R}(t)$, the Euler Angles and the Euler Axis/Angle. In practice, when the smartphone rotates 90°, using the Euler Angle suffers from a singularity which results in the Gimbal Lock problem [6] and significantly pollutes the calculation result. In this paper, we use the Euler Axis/Angle. We define a unit vector $\mathbf{q} = \{\bar{q}^T, q_4\}^T$, where $q_4 = cos\frac{\theta}{2}$, $\bar{q} = \{q_1, q_2, q_3\}^T = sin\frac{\theta}{2}\mathbf{e}$, and $|\bar{q}|^2 + q_4^2 = 1$. According to the rotation process, the rotation matrix $\mathbf{R}$ can be calculated as

$$\mathbf{R} = \begin{bmatrix} q_1^2-q_2^2-q_3^2+q_4^2 & 2(q_1q_2-q_4q_3) & 2(q_1q_3+q_4q_2) \\ 2(q_1q_2+q_4q_3) & -q_1^2+q_2^2-q_3^2+q_4^2 & 2(q_2q_3-q_4q_1) \\ 2(q_3q_1-q_4q_2) & 2(q_3q_2+q_4q_1) & 2q_1^2-q_2^2+q_3^2+q_4^2 \end{bmatrix}.$$

With Equation (4) and (5), we finally have

$$\frac{d\mathbf{q}(t)}{dt} = \frac{1}{2}\begin{bmatrix} q_4(t) & -q_3(t) & q_2(t) \\ q_3(t) & q_4(t) & -q_1(t) \\ -q_2(t) & q_1(t) & q_4(t) \\ -q_1(t) & -q_2(t) & -q_3(t) \end{bmatrix}\begin{bmatrix}\omega_x(t)\\ \omega_y(t)\\ \omega_z(t)\end{bmatrix}. \quad (6)$$

With this non-singularity differential equation, we can produce successive rotations if the initial state $\mathbf{q}(0)$ and the real-time angular velocities $\Omega(t)$ during the rotation are available. In our system implementation, we use the fourth-order Runge-Kutta method to solve Equation (6).

B. ATTITUDE FROM COMPASS AND GRAVITY

The phone attitude in the geo-frame using compass and gravity is represented by the rotation matrix $\mathbf{R}$:

$$\mathbf{R} = \begin{bmatrix} \frac{b^2-ccos\delta}{acos\delta-bsin\delta}\sqrt{K} & sin\delta & a \\ \frac{ac}{acos\delta-ab}\sqrt{K} & cos\delta & b \\ \sqrt{K} & b & c \end{bmatrix}, \quad (7)$$

where $a = \frac{-g_x}{g}$, $b = \frac{-g_y}{g}$, $c = \frac{-g_z}{g}$ and $g = \sqrt{g_x^2+g_y^2+g_z^2}$.

$$K = \frac{((acos\delta-ab)(acos\delta-bsin\delta))^2}{a^2(acos\delta-bsin\delta)^2} + ((b^2-ccos\delta)(acos\delta-ab))^2 + ((acos\delta-ab)(acos\delta-bsin\delta))^2. \quad (8)$$