DirectMe: A Mobile Phone Algorithm for Direction Detection

Alex T. Mariakakis
Duke University, ECE/CS 2013
Advisor: Dr. Romit Roy Choudhury

Contents

1 Abstract

Indoor localization has been one of the forefront topics of mobile computing research in recent years. Many algorithms within the field involve a process known as dead-reckoning; with respect to smartphones, this entails the leveraging of inertial sensors (e.g., accelerometer, gyroscope) in order to capture a person's trajectory over time. Two issues arise when dead-reckoning is used on its own: (1) *Inertial sensors tend to be noisy.* Most dead-reckoning calculations require some sort of integration, so small sensor errors result in the accumulation of large localization errors over time. (2) *The phone must be maintained in a specific orientation, particularly for the use of the compass.* Such a restriction is not conducive to how people normally interact with their smartphones, rendering dead-reckoning impractical. My research is aimed towards loosening this limitation.

I propose DirectMe, a real-time algorithm intended to determine a user's facing direction (UFD) regardless of the phone's orientation by taking advantage of the same intuition that preceded dead-reckoning. People tend to manipulate their smartphones in certain orientations when they are texting, watching a movie, etc; when this is the case, an initial UFD measurement can be made using the compass. As the phone rotates, various inertial sensors can capture the change, thereby updating the UFD measurement over time. I believe this algorithm is a step towards enabling more practicality for indoor localization applications.

2 Introduction

As smartphones have become more ubiquitous in today's society, the demand for more sophisticated mobile applications has grown at an exponential rate. One direction in which many of these applications have developed has been through taking advantage of the phone's internal GPS to achieve building-level localization. For instance, *FourSquare* uses the GPS to narrow a location estimation to within an 8m radius circle and then has the user manually select from a list of nearby buildings [1, 2]. *Sonar: Friends Nearby* does not implicitly deliver building-level localization, but rather uses the GPS to discover other users in close proximity [3]. The next step for applications moving towards this direction is likely going to involve room-level localization, which is similar to building-level localization, only with finer guaranteed accuracy; while a localization error of 8m from the GPS may be tolerable for indicating the correct building, the same error is likely not acceptable for indicating the correct room.

A majority of the current attempts to actualize room-level localization tend to fall into two categories. The first pertains to systems that are primarily based on the utilization of ambient signal processing. *RADAR*, for example, records the signal strength of multiple radio-frequency base stations to trilaterate the user [4]. *SpinLoc* takes advantage of the human body's ability to attenuate WiFi signal strength by having the user spin in a complete circle so that they may be triangulated from nearby access points [5]. Systems like these have been shown to achieve a high amount of accuracy, but at the cost of scalability issues

and unreasonable bootstrapping. Before such signal processing systems can be implemented for a given building, an engineer must wardrive the entire floorplan by walking to every square meter; moreover, for systems that use unstable signals like WiFi, such a process may not be a one-time cost, but rather be necessary whenever system accuracy begins to degrade.

The other category of systems attempting to actualize room-level localization uses a technique called dead-reckoning. In this method, the user's trajectory over time is constantly extrapolated using data from the phone's inertial sensors and some prior knowledge about the user's initial location. *UnLoc*, the project that motivated this paper, uses many of the signal processing techniques mentioned earlier, but eliminates the wardriving overhead by utilizing pure dead-reckoning as a lower bound [6]. Dead-reckoning has been a heavily-researched technique in a variety of applications [6, 7, 8], yet it has faults that have yet to be completely resolved. The problem that this paper addresses is the fact that most of the current dead-reckoning implementations enforce tight restrictions on how the phone must be held in order to properly measure the user's direction of motion. If this calculation is performed using the gyroscope, the phone's axis of rotation and initial compass heading must be known *a priori*. Performing the same calculation with the compass requires that the phone is held flat, as if it were an analog compass; furthermore, the compass is not particularly accurate in indoor environments where the compass can be adversely affected by objects like elevators and water fountains.

The goal of DirectMe is to present an algorithm that incorporates some combination of the phones sensors outside of the compass to provide an estimation of the user's facing direction (UFD) without any delay. This is a particularly difficult problem because the phone is a separate entity from the user, so the orientation between the two must be established before anything can be done. The core concept behind the algorithm design is that many of the components of geographic dead-reckoning correspond to some analogous component with respect to phone orientation, which leads to what I choose to call directional dead-reckoning. In geographic dead-reckoning, the user's initial location and direction must be known. In directional dead-reckoning, this corresponds to knowing the phone's orientation relative the user at times when it is possible. As the user walks around, geographic dead-reckoning uses the accelerometer in some way to measure the distance that the user has traveled and either the compass or the gyroscope to measure the direction. The complement to this in directional dead-reckoning is using the gyroscope to track how the phone has deviated from its initial orientation.

3 Algorithm Design

When using a geographic dead-reckoning scheme, the user's final location is found by combining their initial location with an updated displacement measurement over time. The directional dead-reckoning proposed in this paper requires similar two measurements: (1) the direction in which the user is initially facing and (2) the angle relative to gravity by which the phone has rotated over time. Let θ_0 denote the first and $\Delta\theta(t)$ denote the latter.

The final UFD, $\theta_f(t)$, may then be succinctly expressed as

$$\theta_f(t) = \theta_0 + \Delta\theta(t) \tag{1}$$

The subsections that follow discuss how θ_0 and $\Delta\theta(t)$ are found.

3.1 Finding the Initial UFD

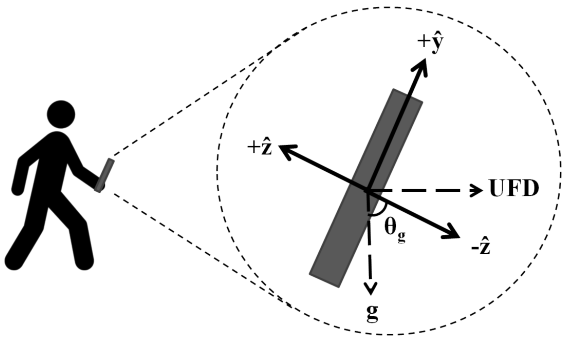

Figure 3.1: A plane of the phone's three-dimensional axis naturally aligning with the UFD.

The compass reading alone is not always a reliable approximation of the user's heading since the device may be at an arbitrary orientation at any given time, making it difficult to ascertain how the compass reading is related to the UFD; despite this, there are instances when the relation may be inferred. When a user interacts with the screen of their phone (e.g., texting, watching a video, playing a game), the device is held in a manner such that two of its coordinate axes are coplanar to the vector that denotes the initial UFD. An illustration of this observation is provided in Figure 3.1. The implication of this observation is that compass readings may be roughly adjusted by integer multiples of 90° based on how the phone is being held by the user, as outlined in the second column of Table 3.1, thereby providing a means of calculating θ_0 in Equation 1. Knowing that two of the phone's coordinate axes are coplanar to the initial UFD not only helps in adjusting the initial compass reading, but also facilitates the use of trigonometry to establish a vector with respect to the phone that represents the forward direction for the user. This vector is compared to the one found in the Section 3.2 to determine $\Delta\theta(t)$ in Equation 1.

Phone Orientation	**Compass Offset**	**Initial UFD Vector**
Portrait	0°	$\langle 0, \cos\theta_g, -\sin\theta_g \rangle$
Landscape	+90°	$\langle \cos\theta_g, 0, -\sin\theta_g \rangle$
Reverse portrait	+180°	$\langle 0, -\cos\theta_g, -\sin\theta_g \rangle$
Reverse landscape	−90°	$\langle -\cos\theta_g, 0, -\sin\theta_g \rangle$

Note: θ_g is the angle between the gravity vector and the $-\hat{z}$-direction.

Table 3.1: The relationship between phone orientation, compass offset, and initial UFD vector.

3.2 Determining Rotation about Gravity

Once the initial UFD has been established, DirectMe no longer leverages the compass to resolve the user's heading, but rather relies entirely upon the phone's inertial sensors to dead-reckon the measurement. The three-axis gyroscope provides a measurement for the phone's rotation about each of the axes individually, with values g_{xi}, g_{yi}, and g_{zi}, as well as a timestamp for the measurement sample, t_i. These values are manipulated to form a instantaneous rotation matrix R_i, which may be multiplied with the old UFD vector $\vec{u}_i$ in order to update it. Below are the equations used to express the raw measurements as a quaternion $\vec{q}_i$, transform it into a rotation matrix R_i (using helper functions provided by the Android API [9]), and then apply it to the old UFD vector.

$$\begin{gathered}\vec{q}_i = \langle w, x, y, z \rangle = \left\langle g_{xi} \sin\left(\frac{\theta_i}{2}\right), g_{yi} \sin\left(\frac{\theta_i}{2}\right), g_{zi} \sin\left(\frac{\theta_i}{2}\right), \cos\left(\frac{\theta_i}{2}\right) \right\rangle, \\ \text{where } \theta_i = \sqrt{g_{xi}^2 + g_{yi}^2 + g_{zi}^2} * (t_i - t_{i-1})\end{gathered} \tag{2}$$

$$R_i(w, x, y, z) = \begin{bmatrix} 1 - 2y^2 - 2z^2 & 2xy - 2zw & 2xz + 2yw \\ 2xz + 2zw & 1 - 2x^2 - 2z^2 & 2yz - 2xw \\ 2xz - 2yw & 2yz + 2xw & 1 - 2x^2 - 2zy^2 \end{bmatrix} \tag{3}$$

$$\vec{u}_{i+1} = R_i \vec{u}_i \tag{4}$$

With the initial UFD vector $\vec{u}_0$ from Section 3.1 and the most recent UFD vector $\vec{u}_t$, $\Delta\theta(t)$ from Equation 1 may be found using their cross product; however, knowing the angle between the two vectors is not enough to properly update the UFD. Both vectors must be orthogonal to gravity in order to be meaningful to the directional dead-reckoning, which motivates the need for more information about the state of the phone.

3.3 Differentiating Modes of Phone Manipulation

DirectMe follows a simple finite-state machine, depicted in Figure 3.2, to determine whether or not the phone's rotation implies a direct effect on the UFD.

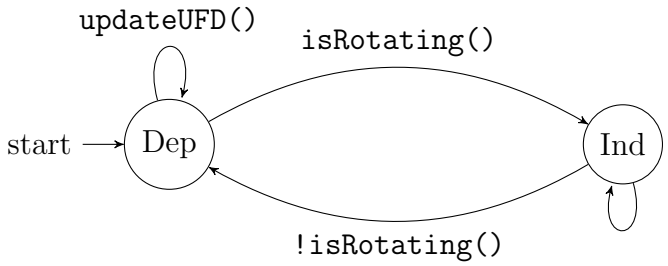

Figure 3.2: Finite-state machine for DirectMe.

The two states the system may be in are the *Dependent* state, when the phone's rotation is directly correlated with that of the user, and the *Independent* state, when the opposite is true; the way in which the state machine transitions between the two is best clarified using a real-life example. Suppose that a user, Alice, begins holding her phone in her hand. The state machine begins in the *Dependent* state because as Alice turns, her phone rotates in an identical manner. Now suppose that Alice decides to place her phone in her pocket. As soon as she begins that motion, the state machine enters the *Independent* state since her phone's rotation is unrelated to her own turning. Once Alice's phone is resting in her pocket, the state machine returns to the *Dependent* state because the way in which she turns will once again dictate the rotation of her phone. To control when the state machine switches between the two states, the `isRotating()` helper method is repeatedly checked. Whenever the phone enters the *Dependent* state, the gravity vector with respect to the phone is stored. A change in the gravity vector of more than some threshold angle is indicative of a significant rotation that is not caused by the user turning, so it will trigger the helper method and transition the state machine to the *Independent* state. Once such a rotation has ended, the state machine re-enters the *Dependent* state and the process repeats.

No matter how the phone is rotating, its overall rotation matrix must be updated using the method from Section 3.2 to track the forward direction with respect to the phone; however, as previously mentioned, not all of the phone's rotations indicate a change in the UFD. For example, if the phone is lifted from directly in front of the user's body towards their face, the gyroscope will note a 90° change, but the user may not have turned at all. Whenever the state machine is in the *Dependent* state, the phone is known to be rotating around an axis that is parallel to gravity, meaning that the user is turning in the same manner and the UFD should be updated using the method called `updateUFD()`. Conversely, when the state machine is in the *Independent* state, the phone's rotation is unrelated to that of the user. Since there is not enough information to say how the user is moving relative to the phone, the UFD is not updated. This is a strong assumption that will be addressed in greater detail in Section 5.1.

3.4 Accounting for Noisy Sensors

The directional dead-reckoning proposed in this paper is no different than geographic dead-reckoning in that its accuracy suffers from the intrinsic error attributed to sensor noise; to alleviate this issue, techniques identical to those applied in geographic dead-reckoning for suppressing error are applied in DirectMe to do the same. One of these techniques is low-pass filtering on all three of the directional acceleration readings, corresponding to the three coordinate axes of the accelerometer. The high frequency noise that appears when the phone rests in the pocket of a walking user tends to overstate the change in acceleration on the phone, so limiting it is desirable; on the other hand, sudden changes in the acceleration when the user rotates the phone need to be tracked as accurately as possible to properly detect how the gravity vector changes. Given Equation 5 for low-pass filtering below, β is a filtering parameter that is meant to strike a balance between the two cases. If the parameter is too high, the filter smooths the data to the point where it lags in response to change; if

the parameter is too low, the filter hardly smooths the data, allowing for the high frequency noise to remain.

$$a_i = \beta a_{i-1} + (1 - \beta) a_{new}, \qquad 0 < \beta < 1 \tag{5}$$

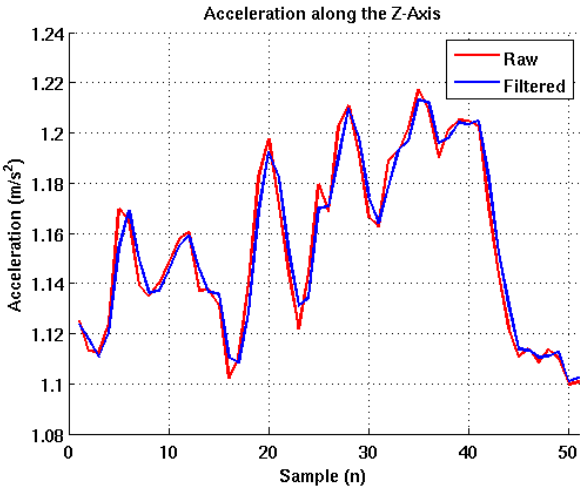

Figure 3.3: Low-pass filtering applied to the accelerometer.

The other signal processing technique applied to DirectMe is a stricter form of low-pass filtering that simply ignores any rotations whose magnitude falls below a certain threshold. Just like the accelerometer, the gyroscope is not ideal and records non-zero values when the phone is at rest. Because the phone's overall rotation is expressed as the accumulated product of matrices, even the slightest values will have an adverse affect on the results, so they are deemed negligible and the change is ignored.

4 Evaluation

DirectMe was implemented as an Android 4.2.x application on a Galaxy Nexus phone for evaluation. Data traces were collected in an office environment without any adjustments to allow for the natural possibility of magnetic interference on the compass. Ground truth headings, which were marked in the room using paper labels, were determined using a minimal amount of map manipulation on Google Earth [10]. The application was tested in three different experiments of increasing difficulty in order to assess its accuracy and robustness. The graphs on the left for the following figures show the results from specific trials, with the horizontal green lines representing ground truth headings. The corresponding graphs on the right are cumulative distribution functions (CDFs) for the error of the measured UFD compared to the ground truth headings. It should be noted that the error only takes into account when the user was facing one of the marked ground truth directions in the office since it is infeasible to interpolate the ground truth for every direction as the user turns.

4.1 Experiment 1: Portrait Mode

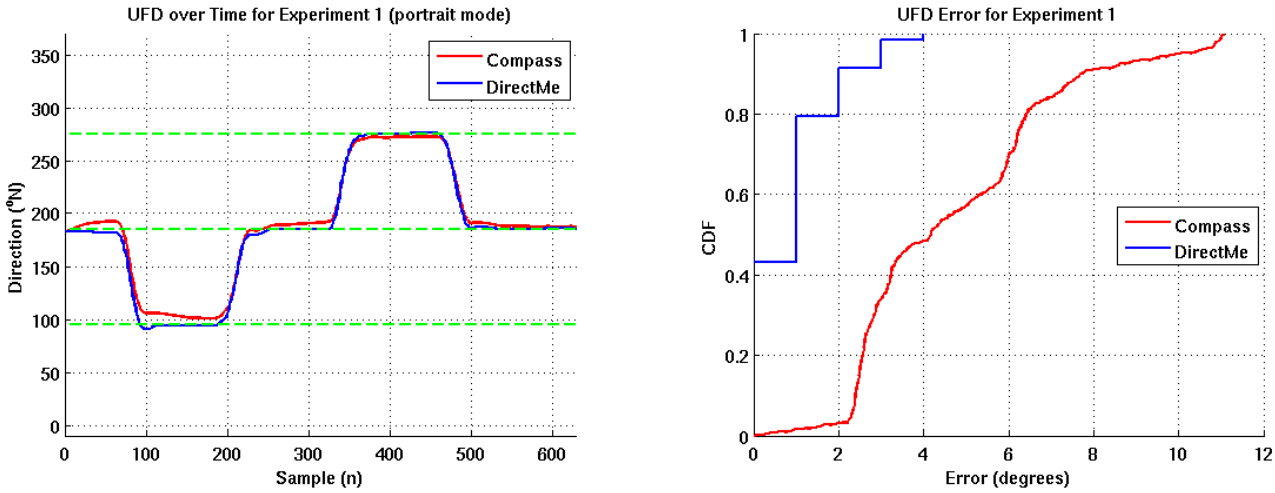

Figure 4.1: (a) UFD tracking compared to the compass for the first experiment. (b) CDF of UFD error with respect to the compass.

In the first experiment, the user held the phone naturally in portrait mode (i.e., vertical screen) so that it was not parallel to the ground, but rather naturally held with a tilt so that the screen was visible to him. He then pressed a button to simulate texting on the screen, which provided an opportunity for the application to establish an initial UFD measurement. The user then turned 90° to the left, back to the original heading, 90° to the right, and then back to the original heading again, waiting at each position for approximately one second.

The results show that DirectMe actually outperformed the compass in multiple aspects. First and foremost, DirectMe tracked the UFD with higher accuracy. The CDF in Figure 4.1b reveals a maximum error of 4° for DirectMe, a large improvement over the maximum error of 11° shown by the compass. Second, DirectMe was much less prone to drift when the user stood still. Ideally, the UFD measurement should not have exhibited any change whenever the user was not turning, so any portion of the graph that seems like it should have no slope should, in fact, have no slope. While there is a small drift in the estimation provided by DirectMe, it is far less noticeable compared to the drift seen in the compass, particularly in the first half of the trial. For instance, as soon as data collection began in the particular trial shown above, the compass reading drifted by approximately 10°, whereas DirectMe drifted by only $2° - 3°$.

4.2 Experiment 2: Landscape Mode

The only difference between the first and second experiments was the fact that the phone was held in landscape mode (i.e., horizontal screen) instead of portrait mode. The goal of this experiment was to confirm that the initial UFD calculations and assumptions were reasonable for different starting orientations. As a reference, the compass trace included in

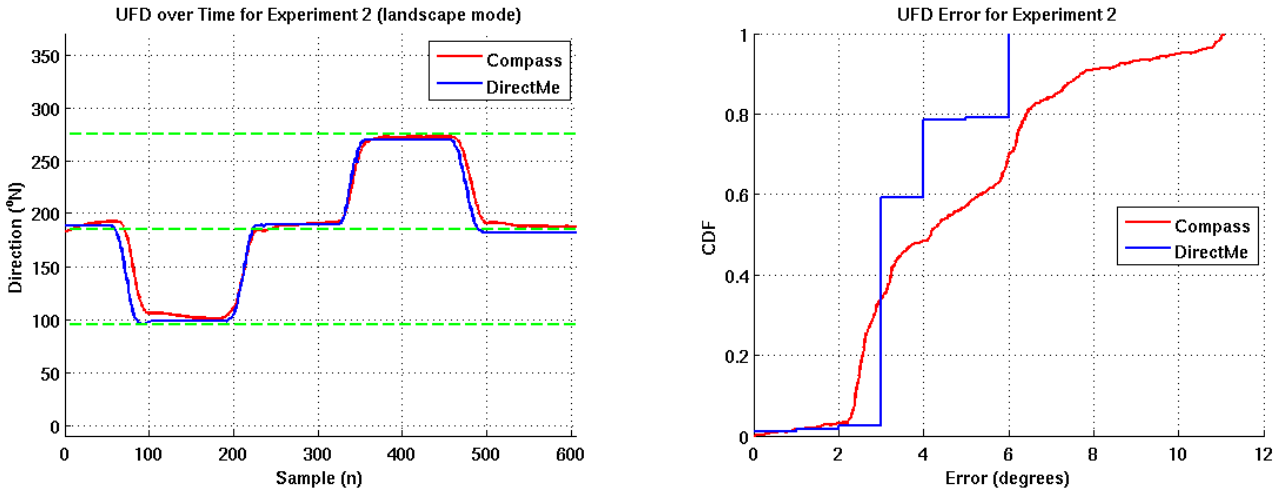

Figure 4.2: (a) UFD tracking compared to the compass for the second experiment. (b) CDF of UFD error with respect to the compass.

Figure 4.2a is just a time-adjusted copy of the one collected from the first experiment.

In terms of accuracy, both the average and maximum error were greater by a few degrees compared to the prior experiment, but not by a significant amount. One possible explanation for this outcome could be the fact that people find it easier to hold their phone perfectly straight in portrait mode than in landscape mode due to the distribution the phone's weight when held in one hand. With respect to the drift exhibited by DirectMe and the compass, the results again show that DirectMe was more steady.

4.3 Experiment 3: Answering a Phone Call

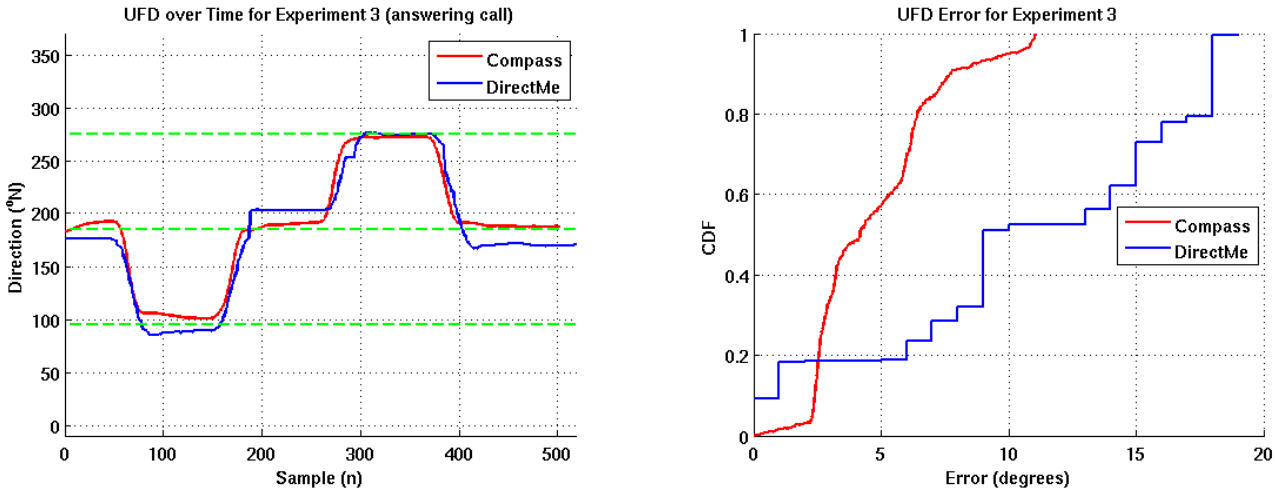

Figure 4.3: (a) UFD tracking compared to the compass for the third experiment. (b) CDF of UFD error with respect to the compass.

The UFD in the first two experiments could have been reasonably estimated using a simple adjustment on the compass, so it was necessary to formulate a third experiment where the compass would have been useless. Just as in the first experiment, the third began with the user holding his phone in portrait mode and tapping a button on the screen. Rather than keeping the phone in that position, however, the user lifted the phone to his ear, as if he was answering a call. Once the phone was next to his ear, the user turned in the same manner as before. The difficulty in tracking the UFD in this case was two-fold: (1) DirectMe had to be able to distinguish which of the two states the phone was in (refer to Figure 3.2) with as much precision as possible. (2) DirectMe had to properly dead-reckon the new vector representing the forward direction for the user during a more complicated movement than a trivial rotation around one vertical axis.

As expected, DirectMe was not as accurate as it was in the first two experiments; the maximum error seen was 18°, while the average error was 10°. Furthermore, drift was more apparent for DirectMe in this experiment, but not beyond what would normally be seen with a compass. While greater accuracy and robustness would have been more desirable, there is no doubt that DirectMe is a vast improvement over the compass in UFD estimation.

5 Limitations and Future Work

5.1 Simultaneous Human and Phone Rotation

The main difficulty inherent in DirectMe lies in the fact that the goal is to determine how one entity, the user, rotates using information from an entirely separate entity, the smartphone. The intuition mentioned in Section 3.1 results in triggers for when the relationship between the two entities can be quantified, but from then on the relationship is purely dependent on the data collected by the phone. There are many scenarios where such a method breaks down. An obvious example is if the user were to place their phone on a table and then walk away. In this case, the user could turn in any way and the phone would never recognize the change. Such a situation could be eliminated by assuming that the phone will always be in direct contact with the user, but that substantially limits the practicality of the algorithm since the user can be in possession of the phone without holding onto it (e.g., phone stowed in a purse).

More complicated examples where the current algorithm fails arise when the phone is being held by the user, but rotates in a different manner. As mentioned in Section 3.3, the assumption is made that when the phone rotates in a manner other than around gravity, the UFD is assumed to be unchanged. This is hardly the case in practice, however, as people are just as likely to place their phone in their pocket before turning a corner as they are to do the same while turning a corner; the UFD would change in the first scenario, but not the second. The algorithm in its current state also fails when the phone rotates around gravity, but the user does not turn at all. Some people find themselves in situations when it is appropriate to move their phone to the side in order to look at the screen (e.g., avoiding glare, talking to someone); DirectMe would be tricked into believing that the user has turned

because the phone has rotated, but the user could actually be sitting in a chair or walking straight. Despite these cases, the aforementioned assumption was made because there is no clear correlation between inertial data on the phone and the turning of the user, so future work needs to be dedicated towards exploring other approaches to remedy this issue.

5.2 Noise Introduction through Loose Handling

For all of the experiments discussed in Section 4, the user held the phone in his hand at all times, but that is hardly the case in real life. In fact, for the majority of the time, people keep their phones in their bags, purses, and pockets. While appearing benign at first, situations like these actually have the potential to introduce a significant amount of noise that could adversely affect the UFD measurement. If the phone is kept in a place where it is restricted from movement, like being held in a hand, there is hardly a problem; otherwise, the phone is free to jostle around, which results in gyroscope readings that are significant enough to pollute the rotation matrix. One might argue that such values should simply be suppressed or filtered in some manner, yet it is important to know the exact moment when the user begins to rotate the phone in order to see if the finite-state machine needs to change, so they cannot be ignored just because they are low values. The algorithm could look ahead at future readings, but the goal of the entire algorithm is to give the exact UFD measurement in real time without any delay. More time should be spent towards investigating when small magnitude rotations should and should not be ignored.

6 Conclusion

Indoor localization has remained to be a heavily researched topic in the field of mobile computing, with an increasing number of proposed solutions emerging that are more accurate than their predecessors; nevertheless, many of the solutions that utilize the inertial sensors on the smartphone restrict the user from holding the phone in a natural manner. To this end, I have proposed DirectMe, an algorithm for determining the direction in which a user is facing no matter how the phone is held. By taking into account some intuitions of how people interact with their smartphones outside of their pockets, I have presented a form of directional dead-reckoning to infer how the phone is oriented relative to the user over time. The DirectMe design hardly leverages the compass, which ensures that the algorithm does not suffer from the inherent drift or reactivity to external magnetic fields that is normally associated with it. Over the course of three different experiments, I have demonstrated that DirectMe in its current implementation can function with an error no greater than 18° and an average of about half that amount. I hope that this work will facilitate the serviceability of indoor localization applications in the future.

7 Acknowledgment

This project would not have been possible without the contributions of numerous people. First and foremost, I would like to thank my advisor, Dr. Romit Roy Choudhury, for his guidance throughout the project. Secondly, I would like to thank everyone in the Systems Networking Research Group (SyNRG) for their advice and camaraderie during my undergraduate research experience. Finally, I would like to thank Associate Dean Martha Absher for her role as coordinator of the Pratt Research Fellows Program.

References

[1] J. Lindqvist, J. Cranshaw, J. Wiese, J. Hong, and J. Zimmerman. I'm the mayor of my house: Examining why people use foursquare - a social-driven location sharing application. In *CHI*, 2011.

[2] S. Gaonkar, J. Li, R. R. Choudhury, L. Cox, and A. Schmidt. Micro-blog: Sharing and querying content through mobile phones and social participation. In *MOBISYS*, 2008.

[3] B. Martin, B. Hargrave, B. Kempe, and R. Chang. Sonar: Friends nearby faq, 2012.

[4] P. Bahl and V. N. Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *INFOCOM*, 2000.

[5] S. Sen, R. R. Choudhury, and S. Nelakuditi. Spinloc: Spin once to know your location. In *HOTMOBILE*, 2012.

[6] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury. No need to war-drive: Unsupervised indoor localization. In *MOBISYS*, 2012.

[7] S. Beauregard and H. Haas. Pedestrian dead reckoning: A basis for personal positioning. In *WPNC*, 2006.

[8] I. Constandache, R. R. Choudhury, and I. Rhee. Towards mobile phone localization without war-driving. In *HOTMOBILE*, 2009.

[9] Google. Android 4.2 api, 2013.

[10] Google. Google earth, 2013.