Distance and Height Measurement using Smartphones

Vigneshwar Padmanaban
Department of Computer Science
Florida State University
Tallahassee, Florida 32306
Email: vp15g@my.fsu.edu

Nithin Tom George
Department of Computer Science
Florida State University
Tallahassee, Florida 32306
Email: ng15j@my.fsu.edu

***Abstract*—Measuring how far an object is away from us or what is the height of an object, a person or a building is normally done with measuring instruments. We have developed an application that can measure the distance and height of objects using your smartphones and no longer have to depend on the measuring instruments. Current apps in the store that does this functionality are inaccurate and has not addressed all the flaws in the method. We are using sensor fusion using complimentary filter to attain more accurate data from the sensors. Basic issue while measuring objects is the users hand movements and position which can vary depending on the users. We have designed a calibration algorithm that will give the average height where a normal user would place his/her phone and how much is the variation in height which the user can make while measuring objects at various scenarios. According to the data collected, it will be grouped to particular groups and data will be fetched from it based on the users activity. Later, we will be focusing on the evaluation under various distances and heights for different users and accuracy of the results obtained.**

I. Introduction

Many of us would have measured the height of another person, the height of a building or how far a point is away from us. We would normally use a measuring instrument for this, a ruler, a meter scale or a measuring tape. It is quite a tedious way to do but what if we have a mobile application that does the same functionality as that of other measuring instruments. So we have developed a mobile application that can measure the distance a particular object is away from us or the height of a building or a person or any object. Accuracy is a key component in this app and there is no room for any errors because a slight change here and there will cause a drastic variation in the final output which is not desirable. Since the current apps in the market are showing lot of errors in the final output we needed to find a way to reduce the errors because with the kind of sensors existing in the market nowadays, if used properly can give better accuracy with negligible error rate. The primary objective of this app is to get the correct attitude of the phone like how is the phone tilted, at what angle because a variation of 1 degree in the output can cause up to 5 cm of error in the final result. For this we are using sensor fusion using complementary filters that fuses the output of all the three IMU sensors, accelerometer, gyroscope and magnetometer. All the three sensors are used for better accuracy because removal of one sensor from the three can cause a lot of errors in the output. The accelerometer gives a gravity vector that points to the center of the earth, the magnetometer works as a compass. A combination of these two can provide the results but it is largely inaccurate because of the presence of noise. A gyroscope alone can give the tilt value and it is far better than the other sensors due to the fact that it responds quickly without any delay. But the disadvantage of gyroscope is it drifts rapidly and the reading can vary depending on the drift. So to avoid the gyroscope drift and noise from accelerometer and magnetometer combination, gyroscope output is applied only for orientation changes in short intervals of time, whereas the accelerometer, magnetometer is used as a support information over long periods of time.

Once the tilt angle of the phone is obtained the next issue to be addressed is the users hand movement while taking the reading and height where the user has placed the phone. A normal user can place the phone at any height and at any distance from his body. So its challenging to find out the exact position of the phone because it varies with different users. But for users of a particular height range, the position of the phone will not vary as such and we can use that logic to estimate the phones height and then generalize it for other users. Another important fact to note is that as the distance from the object increases the users phone position also rises to focus the object. In this scenario, the height of the phone and the distance of the object are directly proportional and the phones height cannot increase after a particular limit which will help in calibrating the data for various users. Once the distance readings are taken, to measure the height of the object user has to focus on top of the object which will cause a rise in his hand position and affect the readings. In order to counter that, readings are taken for a particular user where a user moves his hand while measuring the height and the data has been tabulated for generalization. Thus based on these data collected, issue of users hand movement and phone position issue can be neglected and accuracy can be improved.

All these are valid for users on the same level surface as that of the object. What if the user and the object is at different height level such as a hill or an inclined surface. In that case, we need to find a way to locate the height of both the users above sea level using GPS which will help in getting the desired result. As the normal logic cant be applied on this scenario lot of research has to be done in this area as locating the destination object is a difficult objective to achieve. Once that is attained, we can use the same logic as the one we applied for objects on the same level surface as user.

II. Related Work:

There are a number of related apps that can measure the distance and height of objects but none of them were able to produce a good accuracy. One such app is the Smart Measure which will give the distance and height of objects with error of 15cm to 20 cm. In Smart Measure, the users height is taken directly from the user as input. This will not be accurate because the user will have no idea at how much height he/she is holding the phone. It also does not take into consideration the change in users hand position while measuring different heights and distances which in turn will add to the total error. As the distance from the object increases the users hand position also rises but Smart Measure does not take into account that fact. Similarly for measuring height, the hand position of the user changes while measuring objects at various heights but that variation has been neglected in the app. Also the user will be holding the phone at a particular distance from the body which is not taken to consideration by the app while measuring the distance. These will cause drastic changes in the final output because minute variations in hand position can cause a minimum error of 15 cm. In the app developed by us, we are not asking the user to input the height rather it will be taken automatically by values from the calibration data. Thus it will neglect the error caused due to users hand movement and position leading to better accuracy than the existing apps.

III. System Design

To give you a brief idea on the working of this app, initially when the app is launched, the camera preview with crosshair (for focusing the object) is displayed. There will be two buttons for Distance and Height. First the user has to press the Distance button three times, focusing on the base of the object. This will capture the snapshot of Sensor fusion output and infer the first tilt angle. The user should not move his hand while tapping and remain fixated on the base. If there is a large movement detected between consecutive taps, the app will ask user to tap again. After getting the Distance the user has to press the Height button three times, focusing on the top of the object. This will second snapshot of sensor fusion output, infer second tilt angle and give the Height of object. Here, after each time the tilt angle is calculated, the phone height will be calibrated in a Calibration logic before calculating the Distance or Height of object. So, the main components being Sensor Fusion, Angle, Distance and Height calculation, Calibration1, Calibration2. The reason for three taps is to confirm the user hand is steadily focussed on object reference. This can be replaced with single tap later, but this is the setup we made for developmental and testing purpose.

Another feature present in the app is to include the elevation difference. This is not necessarily the terrain difference, even if the object is present above some known height, and you dont want to include that height, you can ignore it by giving the elevation difference in the app. Even you can give the elevation of user and object, the app will take the difference and include it as adjustment.

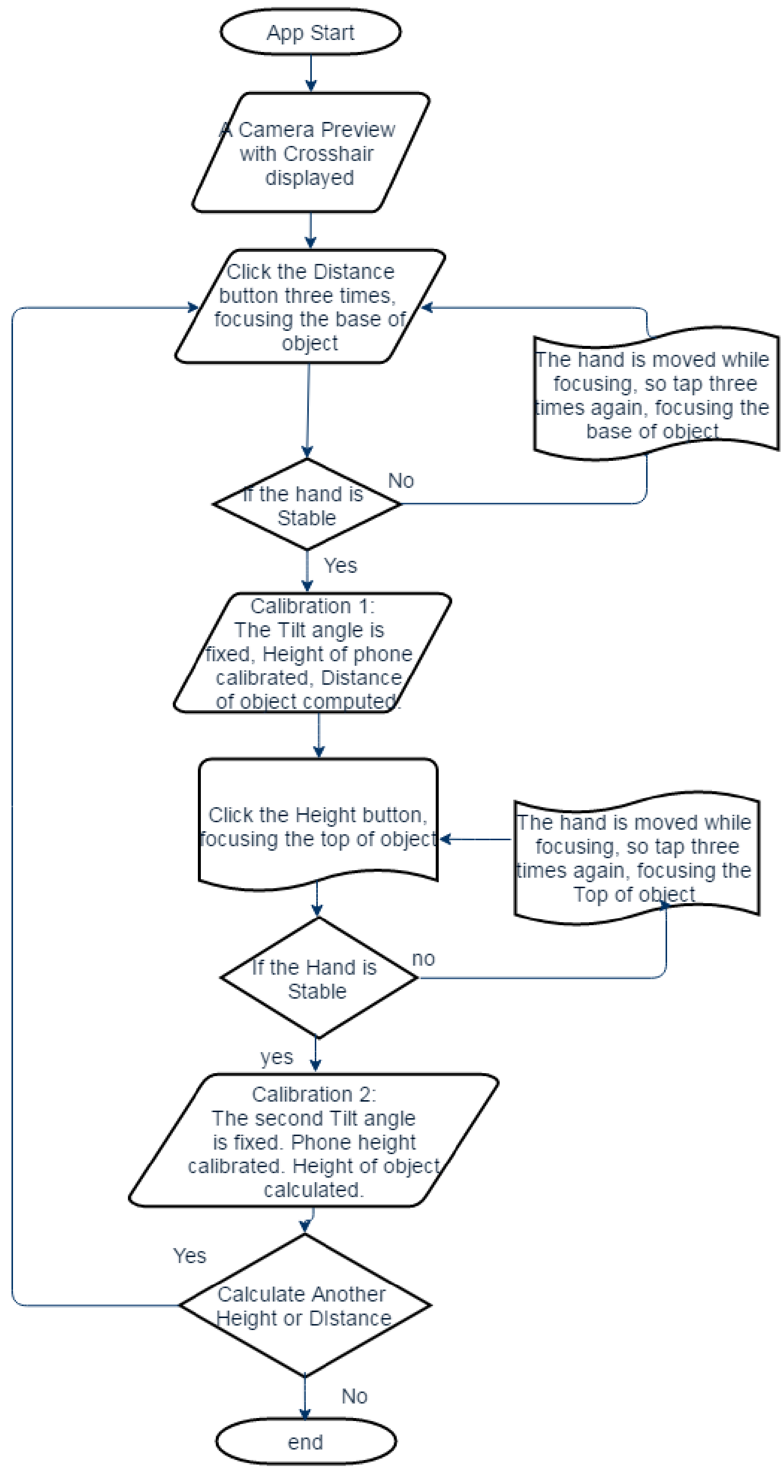


Fig. 1. System Flow

IV. Ground Truth Collection:

The three parameters we have discuss here are: 1. Phone height above surface. 2. Distance between toe and phone. 3. Position of crosshair in display.

A. Phone Height above surface:

This is collected by a suspending a weight (pointed metal object) in a thread as shown below. This is performed by two authors, one holding the phone focusing on object and other person suspend the thread and measures the phone height above ground level (centre of phone to ground).

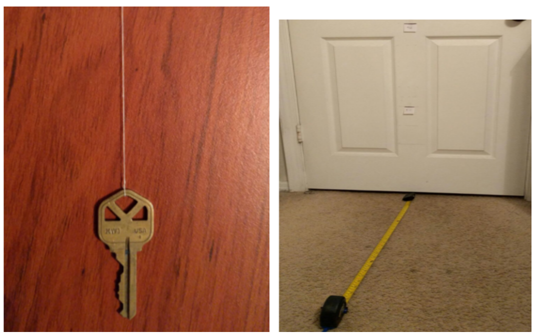


Fig. 2. Suspended weight

B. Toe distance:

Similarly suspending the thread the distance between the phone and toe is noted down. This will be done repeatedly for many distances and height ranges starting form minimum to maximum.

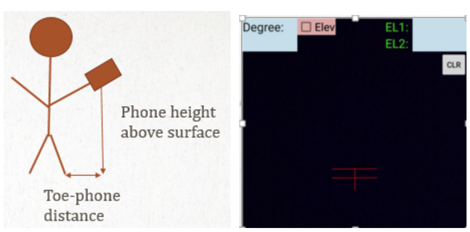


Fig. 3. Phone Height-Toe Distance & Crosshair

C. Position of Crosshair:

In the phone display the position of crosshair plays an important role in getting the accurate tilt angle value. For the same distance from object, the position of cross hair will vary the tilt of phone. We formed an equilateral triangle (Height of phone above ground level = Distance of phone from object), and attempted to get 45 degree inclination in various cross hair positons. We fixed two positions at which the accuracy was desirable. The top line will be used for the distance measurement purposes and bottom line is used for height measurement purposes. Because each of those lines gave best results for respective measurements.

V. Design Details:

A. Sensor Fusion:

The IMU sensors like Accelerometers, magnetometer and gyroscope are available in almost all smart phones. For our purpose of calculating inclination, either the accelerometer alone or gyroscope alone can be put to work. But they provide a bad performance individually. When they put into work combined, they can provide a smooth output. The accelerometer and magnetometer combination is prone to noise. Even when the phone is kept in a stationary place. When in hand the noise levels are very undesirable. In case of gyroscope the output is very smooth and desirable for a short time. But, when used for a long time, they tend to drift. So the concept of sensor fusion, combing all these sensors through complimentary filter is adopted. This is same as low pass filtering the Accelerometer-Magnetometer output and high pass filtering the gyroscope output. Whenever there is an orientation of phone, the gyroscope output will be followed and during less or no orientation, the accelerometer output will be followed. In this way, we are eliminating the accelerometer noise and also the drifting problem of gyroscope. [2] [1]

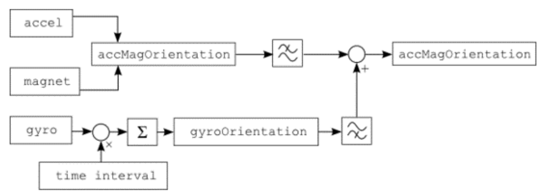


Fig. 4. Sensor fusion

B. Pythagoras Theorem:

In order to find the distance and height the basic principle used is Pythagoras Theorem. We will be using different variations of this principle in various scenarios to get the correct measurements. The basic measurement which the app will do is measuring how far a particular object is away from us. In order for that to happen we have to know how much the phone is inclined, i.e , the tilt of the phone. Using sensor fusion of the three IMU sensors and complementary filter we will be getting the angle of inclination. Once we get the inclined angle then our focus is on getting the phone height from the distance calibration data. Data is fetched from the distance calibration by using the tilt angle got in the previous step. As soon as we get the phone height then distance is calculated based on the Pythagoras theorem. [4]

1) Distance Calculation: As shown in the figure 1, to measure the distance first the crosshairs must be focused at the base of the object. By sensor fusion and complimentary filter the angle of inclination 1(the angle at which the phone is tilted) is found. From the distance calibration data, we get the users phone height shown by h in the figure 1. Then we get the distance d from the object as:

$$tan\theta 1 = d/h => d = h * tan\theta 1 \quad (1)$$

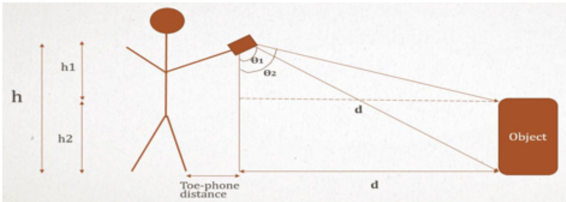


Fig. 5. Distance and Height logic

But this is not the actual distance of the object, as shown in the figure there is a toe distance factor that has to be added to the calculation to get the exact distance. The toe distance will be fetched from the distance calibration data table. Therefore the exact distance of the object is:

$$Exactdistance = h * tan\theta 1 + toe - phonedistance. \quad (2)$$

2) Height Calculation: After getting the distance of the object, the user focusses to the top of the object to get the height. The object can be a person, a building etc. As the user focus to the top of the object his position of the phone could vary and is dependent on the distance from the object and the angle theta2 at which the phone is held. Based on these values, a new user height H is fetched from the height calibration. Using the value d which is the distance of the object and inclination of the phone theta2 the height h1 can be found as shown in figure 1. Then the height of the object will be the difference between the new phone height H and the calculated value h1. Therefore h2 = H h1 where h2 is the object height.

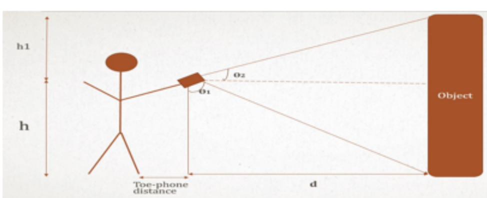


Fig. 6. Measuring objects above user height

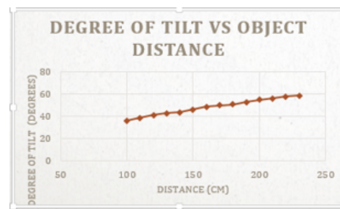
Suppose the object height is more than the phone height then we need to add a new logic than the one just mentioned. First we need to get the tilt angle theta2 at which the phone is inclined while focusing at the top of the object as shown in Figure 2. Based on the value of theta2 and distance d, height h1 can be found. Now the new height h1 is added to the phone height calculated from the Height calibration data H to get the object height.

$$tanjj2 = d/h1 => h1 = d/tanjj2 \quad (3)$$

Therefore the Height of the object = H + h1. Thus we are able to attain height of objects whose height is below the phone height and above the phone height. If the user wants to measure another height without going to the initial state he can focus to the new height and get the height of object, as the distance remains the same. In this way, multiple heights can be obtained and rather quickly.

C. Calibration 1: Distance:

The first part of the app is to calculate the distance of object. Then only the height of object can be estimated. The main issue in inaccuracy of the distance measurements is the varying hand position of the user, while holding the phone and measuring different distances. The height of phone above surface will linearly increase for increased distances and will become close to stable for farther distances. This height variations of the phone is recorded for various distances, using the procedure explained in the Ground truth collection. Below is the snippet of data and corresponding graphs. As



Distance	Phone_Height	Toe_Distance	Degree
100	122	18	35.8
110	122.5	17	38.37
120	122.8	19	40.9
130	126	20	42.51
140	128.1	21	43.6
150	127.8	20	45.72
160	130	19.5	48.39
170	133.1	21.5	49.55
180	133.5	22	50.68
190	135.2	20	52.4
200	136.2	21	54.68
210	137.5	19	55.72
220	138.5	17.8	57.52
230	139.1	18.9	58.39
240	139	21	58.76
250	143	21.5	59.6

Fig. 7. Snippet of data and Graph

you can see above, for various distances starting from 100 cm up to 360 cm, we recorded the following parameters: Phone height above surface, distance of phone from toe and Degree of inclination displayed in the phone. As reflected in the graph, with increase in the distance of object, the phone height increases linearly and degree of inclination also increases linearly. The toe distance is random, as it follows a semi-circle path when measuring distances from shortest to largest. We took an average value for toe distance calibration. For phone height, we introduced a logic to select phone height dynamically, based on the angle of inclination. For example, say for the angle of inclination of 38 to 41 degrees, we used the phone height of 122.8 cm. Similarly we chose for other ranges.

D. Calibration 2: Height:

During the height calculation, the user hand position changes (increases) while trying to focus the top of the object. This causes error in the height calculation. So, similar to distance calibration, the height of phone above surface is measured, for various object heights and at various distances.

Distance	Height	Phone_Height	Degree
70	0	119	24.9
70	30	123.1	32.8
70	40	125.04	35.78
70	50	126.98	38.76
70	60	128.92	41.74
70	70	130.86	44.72
70	80	132.8	47.7
70	90	135.84	52.5
70	100	138.88	57.3
70	110	141.92	62.1
70	120	144.96	66.9
70	130	[illegible]	71.7

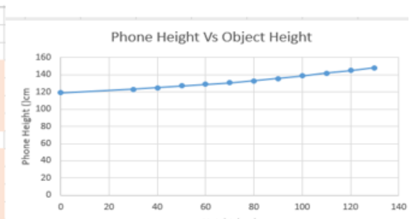


Fig. 8. Snippet of Ground truth data and Graph

For same measuring the same object at various distances, the angle of inclination is varying. As you can see below a snippet of data is shown and its corresponding graph. For the same distance, increase in the object height causes increase in the phone height also increase in the degree of inclination. As evident from graph, the phone height and

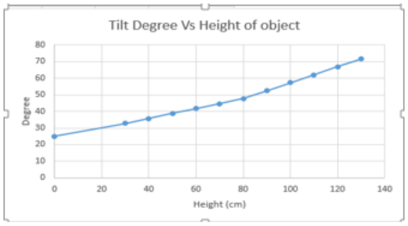


Fig. 9. Degree Vs Height

degree increases linearly, with increase in the height of object. So, the calibration logic here is based on the combination of degree range and distance range. For example, for a distance of 65 to 75 cm, and for degree of inclination ranging from 41 to 44 degree, the phone height selected will be 130.86 cm. The range of distance and degree was selected based on the best results obtained from the evaluation. Similarly for the second scenario of measuring heights above the user height, the phone height will be exceeding the users total height. In this case, a separate calibration data is needed, to correctly guess the phone height above surface.

VI. EVALUATION:

A. Asserting the root cause of error:

The error caused in distance and height measurement is mainly due to the user hand is freely moving up & down and towards & away from the body. This is because of the natural action of human hand while focusing on the objects at various heights and distances. To prove this we proposed another way to hold the phone, in which the user has to hold the phone in both hands, with wrist being rested below the chest. By this, the user cannot move his hands in any manner, except the tilting of phone action. Using this way, we can see that, the app is performing extremely well with distance and height measurements are well within the 3 cm error, even while measuring at distant ranges like 300 400 cm. This proof ascertains the fact that the user hand motion is the main reason in giving large errors like 15 -25 cm. The drawback in this approach is, the user cannot measure heights above his head, as this restriction made it difficult to view the phone screen and also users feel high discomfort in this position. The alternative approach can be to use a tripod stand (provided that user should input the height of tripod to app). But, our intention is to reduce the error while the user casually uses the app in hand, so the distance and height calibration is performed.

B. Distance evaluation:

We evaluated the performance of the app by measuring various distances, starting from lowest to highest and calculated the error rate. As you can see below, the snippet of evaluation data is shown along with the graph. The actual distance Vs Obtained distance is shown below. The mean error is 4.7 cm, with lowest error being 0.7 cm and highest error being 8 cm, while measuring within the range of 450 cm. The error rate is spiking rapidly above this specified range.

Distance	Output	Degree	Error
90	91.9	30.78	-1.90
100	102.1	34.1	-2.10
110	109.3	36.3	0.70
120	114.6	37.8	5.40
130	122.1	39.9	7.90
140	138.2	42.8	1.80
150	154.4	44.6	-4.40
160	155.2	46.2	4.80
170	164.8	48.2	5.20
180	173.9	49.2	6.10
190	187.2	51.1	2.80
200	193.1	52.1	6.90
210	204.6	53.6	5.40

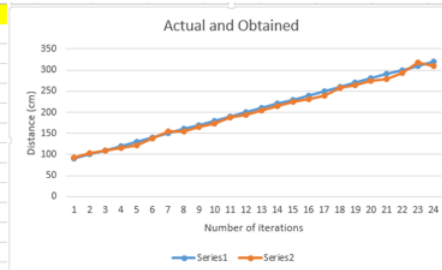


Fig. 10. Distance evaluation

C. Height evaluation:

The performance of app in measuring the various height is shown below. As you can see the mean error being 4.3 cm, the lowest error is 1 cm and highest error being 7.5 cm, while measuring a range of height from 30 cm to 130 cm, while measuring at a distance ranging from 80 to 400 cm.

ACTUAL	OBTAINED
30	34.4
40	41
50	46.3
70	72.9
75	77.1
80	78.9
90	86.8
95	92.2
100	95.9
110	105.5
120	114.1
130	122.5

Fig. 11. Height evaluation

VII. POINTS OF DISCUSSION

The accuracy of this app is still around 4 cm. With this accuracy, the usability of app is restricted to users who do not care about getting centimetre level accuracy. But in the real world environment always need accurate value. This can be due to the inaccuracy in the ground truth data collected manually using thread and weight. Also, its quite difficult to exactly focus the cross hair in the base of the object or top of object.

VIII. FUTURE WORK

In order to further improve the accuracy, the ground truth data can be gathered with more reliable method of recording phone heights and toe distance. Either an instrumental method or image processing method. For focussing objects perfectly, the camera zoom feature can be leveraged. Even more effectively, an image processing algorithm can be implied to correctly fix the cross hair to base/top of the object, after taking an image. This will correct any degree errors happened while focusing. These suggestions are reserved for future.

The other problem of elevation difference between user and object, can be addressed by knowing the difference and using that, we can adjust the height value. Currently the app manually getting the two elevations, calculate the difference correct height will be calculated. This can be automated by using GPS to retrieve the elevation data at two points. The challenge in this feature is to get the elevation details at the object point, given the latitude and longitude of the user position. There are many algorithms for determining this, based on the need, whether short or long distance. This has been reserved for future work. [7]

IX. Conclusion

The auto calibration of phone height while measuring distances and height, has improved the performance of app drastically, rather than giving a random phone height value (current apps in market dependent on user to input phone height). The proposed app will get the user height as input (most of the users know their height) and re-calibrate the calibration data. So that it avoids the users effort to measure the phone height above surface. By getting calibration data from users of various height ranges, this app can be calibrated to the best.

Acknowledgment

This Research has been performed as part of the CIS5930: Wireless and Mobile Computing Course Project, Spring 2016.

References

[1] Pengfei Zhou, Mo Li, Guobin Shen,*Use it free: Instantly knowing your phone Attitude*

[2] Sensor Fusion Tutorial ,*http://plaw.info/2012/03/android-sensor-fusion-tutorial/comment-page-1/*

[3] Matlab Android Interface ,*http://blogs.mathworks.com/community/2014/10/06/acquire-data-from-device-sensors-with-matlab-mobile/*

[4] Using Accelerometer for Inclination Sensing ,*http://www.digikey.com/en/articles/techzone/2011/may/using-an-accelerometer-for-inclination-sensing*

[5] Complimentary filter Reading , *http://www.pieter-jan.com/node/11*

[6] Low pass filtering Accelerometer data,*https://www.built.io/blog/2013/05/applying-low-pass-filter-to-android-sensors-readings/*

[7] Google maps Elevation API, *https://developers.google.com/maps/documentation/elevation/intro*

[8] Donn Felker, *Android Application development for dummies*