Remote object localization using Android devices

Jacob Luebbers
Southeast Missouri State University
jpluebbers1s@semo.edu

Katherine Tynan
Truman State University
ktynan@truman.edu

Nickolas Wergeles
University of Central Missouri
nmw96790@ucmo.edu

Hyun Jang
Graduate Mentor
University of Missouri
hjang@mail.missouri.edu

Aleksandre Lobzhanidze
Graduate Mentor
University of Missouri
agl7dd@mail.mizzou.edu

Qia Wang
Graduate Mentor
University of Missouri
qwch9@mail.mizzou.edu

Dale Musser
Faculty Mentor
University of Missouri
musserda@missouri.edu

Yi Shang
Faculty Mentor
University of Missouri
shangy@missouri.edu

Abstract

Given the prevalence of Android devices in modern society, there are a large number of practical applications for the mobile capability of positioning a remote object. The purpose of this research project is to explore the feasibility of a system for locating a remote object using angle-based triangulation and to compare that to results obtained from previously implemented image-based methods after they are ported to Android from the computer. The angle-based implementation is done using multiple sensors on an Android device, namely the GPS and compass. There are two image-based methods currently implemented that were ported to Android. The two methods consist of a one image and a two image that both use cropping techniques and the OpenCV library in their implementation. Testing of the three methods is accomplished by attempting to pinpoint the location of an object using a variety of test points that exhibit numerous possible scenarios. Preliminary test results are incomplete and inconclusive, so additional testing will be conducted in order to properly draw conclusions.

I. Introduction

The ability to locate a remote object can be extremely useful in many professional and recreational activities. A mobile application that is capable of doing so could be used to find the location of landmarks while on vacation, animals while hunting, or opposing troops during war. Work on developing

an application that can do so has already been done by our research group, focusing on an image-based implementation. Thus far, one image and two image-based methods have been developed on the PC, but only the one image method has been successfully ported to Android. It is with that method that the method developed in this project will be compared. This project focuses on the development of an angle-based implementation of the problem for Android to test for the feasibility and accuracy of such a method. Its accuracy is compared to the test results that the one image-based method returns, and, in the future, will be further compared to the two image-based results should the port to Android be completed.

II. Method Implementation

The algorithm used to calculate the location of the remote object uses a combination of trigonometry and algebra to utilize the idea of intersecting lines. First, the slope of the lines formed by the compass reading is calculated. In order to do so, the Cartesian angle equivalent to the compass reading is found. This is done by subtracting the compass reading from 90 if it falls in the first three quadrants and from 450 if it is in the fourth quadrant, resulting in the necessary angle measures for trigonometric equations. The slope is then found using the equation:

$$m = \tan(\theta) \quad (4)$$

where m is the slope of the line and θ is the measure of the angle. Once the slope is found, the y-intercept of the line can be found using that slope and the GPS coordinate of the data set. This is done by rearranging the slope-intercept form of a line to:

$$b = -x * m + y$$

where b is the y-intercept, m is the slope, x corresponds to the longitude of the point, and y corresponds to the latitude of the point. Now that both y-intercepts are known, the location of the object can be found by setting the two equations equal to each other and finding the intersection. This is accomplished by rearranging the equation obtained by that, with the result being:

$$x = (b_1 - b_2)/(m_2 - m_1)$$

with x being the longitude of the object, the b's are the y-intercepts, and the m's are the slopes. Once this is found, all that remains to be done is finding the latitude of the object. This is found by simply plugging the longitude of the object into the slope-intercept form of the line created by the data sets. The result of this is:

$$y = m * x + b$$

where m and b are the slope and y-intercept from one of the lines, y is the latitude of the object, and x is the longitude of the object.

The method relies heavily on the sensors present in nearly all Android devices, namely the GPS and the digital compass. More specifically, it uses the getOrientation string of methods to get the compass readings and the getLastKnownLocation method in the locationManager class. Two sets of data consisting of a GPS coordinate and the compass heading to the remote object are necessary for the localization. There

are problems inherent with the sensors that need to be resolved to get adequate data from them. First, the digital compass uses magnetic fields in order to determine magnetic north. This registers a large amount of ambient magnetic noise that results in a rapidly changing value that has a significant amount of variance. To best amend this issue, it was resolved to implement a running array that takes in each sensor reading and continually shifts out the oldest reading. The average of this array is calculated each time a new sensor reading is obtained and that average is what is outputted to the screen, the final result of which is a significantly more stable reading. Also, the readings from the digital compass are for magnetic north rather than true north, so the difference between magnetic and true north needs to be compensated for. This is achieved by using the built-in function, getDeclination. This is a function in the GeomagneticField class, which contains the GPS coordinates, altitude, and time of the GPS fix. The getDeclination function uses those members of the class to calculate the difference between magnetic and true north and this value is added to the angle given by the compass, which results in a compass heading relative to true north.

Once the location of the object is known, all that remains to do is plot the points onto a map to show the location of all three points. This is done by creating a new activity that has only a map object in the view and plots overlays onto the map. The data is also put into a text file for future reference.

III. Experimental Method

Testing of the angle-based method is accomplished using a systematic set of distance and angle combinations. The experiment is set up in an open area with a central object that allows for a multitude of positions at consistent ranges located all around the object, ideally encircling it. A range finder is used to find the distance from an object to the phone capturing the data. Ranges of 10, 25, 50, and 75 yards are used, with 3 distinct data sets being taken, two of which had acute angles at the point of intersection and the last having an obtuse angle. In addition to those tests, additional test points were taken where the points were not equidistant from the object and the angle formed by the line connecting the two points and the compass heading from point 1 to the object is obtuse. For each distance or angle combination, two data points must be captured in order for the program to calculate the location of the object. The error is calculated by comparing the distance between the points given by GPS to the actual location of the object and the location the program returned and dividing by the true distance.

IV. Results

The experiments resulted in an average of 53.8% error in terms of distance between the points. There was a very wide range of values of error, from 1.7% to 328.4%. Figure 1 shows the error for all the data sets, namely the error in distance between point 1 and the object, point 2 and the object, and the average of the two. It shows that other than a few data sets, they are mostly consistent. Interestingly enough, the drastically inaccurate data sets all fall into one category of tests, which is where the two

lines met to form an obtuse angle at the object. When these are removed to make a graph of only data of sets forming acute angles, it looks much different, as shown in figure 2 and the average error drops to 37.29%.

In comparison, the one image-based method gives back very accurate results on the whole. As shown in figure 3, the results are fairly consistent and only the worst tests resulted in error of over 20%. The average error produced by the one image-based method is 10.96%, with the minimum error being 2.04% and the maximum being 25.75%.

V. Conclusion

On the whole, the one image-based method is much more accurate than the angle-based method. The one downside of the image-based method is the necessity of knowing the physical size of the object. Since this is frequently unknown, it has a limited range of effective applications. If the size is known, it produces good results, but if not, it is impractical, so the angle-based method would be the better choice in those instances.

Due to the fact that the two most inaccurate test results coincided with the instances where the accuracy returned by the GPS was also very low, it is probable that some of the error is caused by the sensors used in the data collection. Data is collected from each sensor twice for each data set, so the error caused by the sensors is compounded upon themselves, which leads to a drastic increase in the overall error. The accuracy of the digital compass is unable to be ascertained by the equipment available, so its effect on the error of the results is uncertain, but it could have a significant effect as well.

Since the image-based method produces more accurate results, it should be used whenever possible, namely, in the circumstances where the physical size of the object is known. Since it is useless when the size is unknown, the presence of another method is necessary in order for any sort of application to be practical in most circumstances. There is a working two image-based method on the PC, but it has not been successfully ported to Android at the time of writing, so the angle-based method is the best alternative available. So, even though the angle-based implementation is not consistently accurate, it would be good to have it as an alternative whenever the image-based method is unusable.

VI. Future Work

The next step in the development of the angle-based method is to add the functionality to share information between two phones to calculate the object's location. This will allow for a greater level of flexibility in its usage due to the ability for simultaneous data collection. Additional ideas should be pursued that could potentially remove some of the data read in from the sensors to help eliminate some of the error. For example, a method could be explored that gets angles relative to the other point directly rather than needing two GPS points to calculate the angles. Achieving this would lead to only needing one GPS reading, which could decrease the error in the method. Also, more work should be done in completing the port of the two image-based method to Android. If it is successfully ported and capable of producing timely and

consistently accurate results, it should replace the angle-based method due to the level of inaccuracy present in its results.

VII. References

(1) http://code.google.com/android/

(2) http://developer.android.com/index.html

(3) http://stackoverflow.com/

(4) http://www.brightstorm.com/math/trigonometry/advanced-trigonometry/angle-inclination-of-a-line

VIII. Graphs

Figure 1

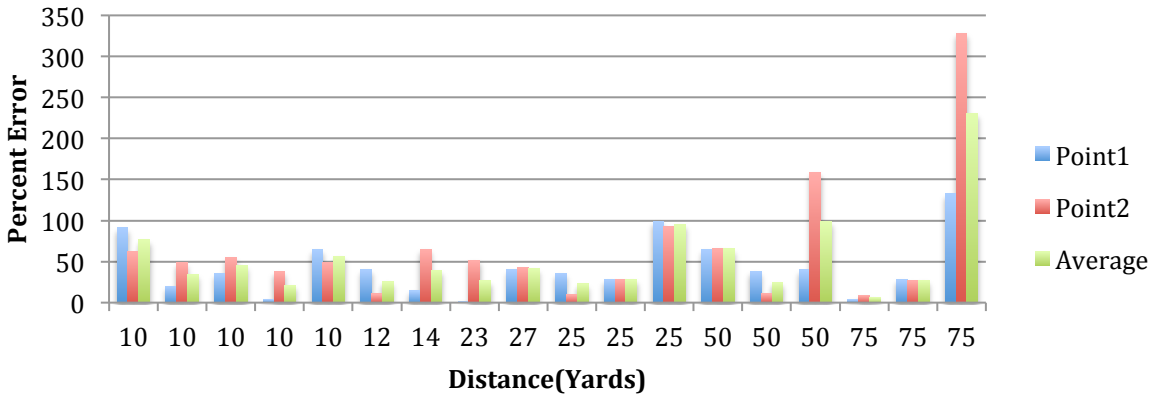

Figure 2

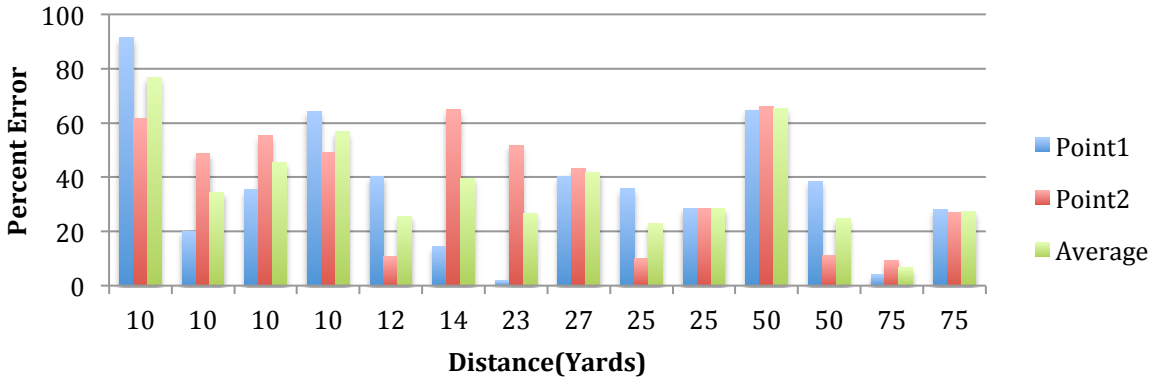

Figure 3