ACCURATE DISTANCE CALCULATION USING GPS WHILE PERFORMING LOW SPEED ACTIVITY

by

BENJAMIN BENNETT

A THESIS

Presented to the Department of Computer and Information Science
and the Graduate School of the University of Oregon
in partial fulfillment of the requirements
for the degree of
Master of Science

June 2018

THESIS APPROVAL PAGE

Student: Benjamin Bennett

Title: Accurate Distance Calculation Using GPS While Performing Low Speed Activity

This thesis has been accepted and approved in partial fulfillment of the requirements for the Master of Science degree in the Department of Computer and Information Science by:

Stephen Fickas	Chairperson

and

Sara D. Hodges	Interim Vice Provost and Dean of the Graduate School

Original approval signatures are on file with the University of Oregon Graduate School.

Degree awarded June 2018

THESIS ABSTRACT

Benjamin Bennett

Master of Science

Department of Computer and Information Science

June 2018

Title: Accurate Distance Calculation Using GPS While Performing Low Speed Activity

In the last 10 years GPS technology has become widely available due to the proliferation of smart devices with GPS capability. GPS was introduced as a method to assist in location and navigation and is still the most common use for this technology today. Many may not consider that GPS can be used for a variety of different purposes. GPS technology can be used to calculate the distance of activity (running, walking, biking etc.) to build applications to promote exercise and a healthy lifestyle. Calculating this distance accurately is challenging due to the amount of error in GPS location measurements and the low speed of many activities. In this thesis I present methods of calculating distance traveled that reduces this error to produce an accurate distance calculation. I also present an application that uses this distance calculation to help promote children to become more active.

CURRICULUM VITAE

NAME OF AUTHOR: Benjamin Bennett

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon, Eugene
Southern Illinois University Edwardsville, Edwardsville

DEGREES AWARDED:

Master of Science, Computer and Information Science, 2018, University of Oregon
Bachelor of Science, Computer Science, 2014, Southern Illinois University Edwardsville

PROFESSIONAL EXPERIENCE:

Graduate Employee – Teaching, University of Oregon, Eugene, Oregon 2017-2018
Software Engineer, Tapestry Solutions: A Boeing Company, St. Louis, Missouri, 2014-2016

ACKNOWLEDGMENTS

I would like to thank Professor Fickas. He was always available to offer guidance and help come up with solutions to any problem I had. I would also like to thank Jason Prideaux for offering valuable input through the project. Special thanks to all the faculty and community members for their insightful feedback. Finally, I must express my gratitude to my family for always providing support, and Natalia, my rubber duck.

TABLE OF CONTENTS

Chapter	Page

LIST OF FIGURES

LIST OF FORMULAS

LIST OF ALGORITHMS

CHAPTER I

INTRODUCTION

1.1 How GPS Works

The Global Positioning System (GPS) was developed and launched by the United States government in the 1970s for use by the military. In the 1980s it was made available for civilian use. The goal of GPS is to allow a device to accurately locate itself from anywhere on Earth at any time. To achieve this goal, GPS consists of a series of 24 satellites orbiting the Earth in a pattern such that at any time and from any location at least 4 satellites will be electronically visible (Hofmann-Wellenhof, Lichtenegger, & Collins, 2012, p. 4). Each GPS satellite continuously broadcast a signal with information about its location as well as its current clock time. GPS receivers listen to this broadcast and through calculating the current location of the satellites via propagation delay and triangulation, its location relative to the satellites can be calculated. This location can then be transformed into a latitude and longitude coordinate on Earth. There is inherent error in the location calculation, but under optimal conditions the location calculated will be within 5 meters of the actual location of the receiver.

1.2 Sources of Error for GPS Location Calculation

There are many sources of error that can occur when transmitting a GPS signal from the satellites to the receivers. Much of this error can be accounted for, such as clock drift of the satellites and signal propagation delay in the upper atmosphere (Grewal, Weill, & Andrews, 2007, p. 103-130). To account for the clock drift, the satellites will periodically synchronize their clock with a common clock. The small drift between synchronizations will have a negligible effect on the overall location calculation. The signal propagation delay through the upper atmosphere can be accounted for by calculating the typical signal propagation delay based on the position of the sun relative to the rotation of the Earth. Given that conditions in the upper atmosphere rarely change unpredictably, this delay can be calculated with high accuracy.

There are some sources of error that cannot be accounted for in the location calculation. This error is the cause of the 5 meters of uncertainty in the location calculation, even under optimal conditions. The primary sources of this error are signal propagation in the lower atmosphere and signal multipath delay. Signal propagation delay in the lower atmosphere occurs when the signal is slowed by bumping into particles in the air such as water particles in clouds. As the weather changes so does the signal propagation delay making it impossible to account for. The largest source of error comes from multipath delay. Multipath delay occurs when the signal bounces off objects before arriving at the receiver. As it is impossible to determine how many times the signal has bounced and off what types of objects, there is no way to calculate the delay. Multipath delay is most noticeable when the receiver is near a building, leading to a large variation in the location calculation between readings. The error will typically be much larger than the 5 meters of error as is typical under optimal conditions.

1.3 Previous Work

There has been much work in improving the accuracy of GPS location calculation while walking using both GPS and an Internal Navigation System (INS) (Cho, Mun, Lee, Kaiser, & Gerla, 2010; Eliasson, 2014; Godha, Lachapelle, & Cannon, 2006). This worked by attaching a INS device to a person, often on their foot, and using a Kalman filter combining the data from the INS device with the GPS coordinate to better determine the actual position of the person. The purpose of the INS device was to determine when the person was taking a step. With previous knowledge of the test subjects stride length, the GPS coordinate received could be more accurately calculated.

Another study tested the use of a Kalman filter to increase the accuracy of GPS coordinate measurements (Eliasson, 2014). A Kalman filter is an algorithm used to better estimate the actual value of data reading that contains inaccuracies. It does this by looking at the measurements over time to make a prediction of the current state. In the case of GPS, the current state is the location of the receiver. The Kalman filter averages the predicted state and the measured state to estimate a more accurate location measurement. The average is weighted based on the known noise level of the reading. In the case of GPS, this noise level is calculated by the receiver for each location measurement. In this study, Eliasson compared the overall distance measured using the Kalman filter against using an averaging method, similar to the algorithm described in section 2.3. They found that there was little difference between the averaging method and the Kalman filter.

CHAPTER II

GPS DISTANCE CALCULATION METHODS

Four methods of calculating distance from GPS data were evaluated in this study. A key property of each method is their ability to update the distance calculation in real time along with the ability to track a wide variety of activity. The intended use of the distance calculation is to calculate distance of low speed activity. Low speed meaning any activity that is not motor assisted. These are activities such as waking, biking, jogging, etc. Many previous methods for calculating distance relied on using an INS device. The use of the INS device to measure steps would not allow activities such as biking.

The four methods chosen were; calculating distance between raw GPS readings, calculating distance based on the speed value provided by the GPS receiver along with the time between GPS updates, averaging GPS readings together into a single point, and taking the line of best fit of a collection of GPS readings and calculating the distance along that line. Each of these methods will be explained in detail in sections 2.1, 2.2, 2.3, and 2.4 respectively.

2.1 Raw GPS Data

The first method evaluated was calculating distance from raw GPS data. Raw GPS data is the latitude and longitude value determined by the GPS system on a device with no filtering or modification of this value. The total distance is calculated by calculating the distance between consecutive GPS coordinates and summing these distances. The distance between two GPS coordinates is calculated using the Haversine formula for measuring the distance between two points on a sphere (Chopde & Nichat, 2013). See Formula 1.

Formula 1 Haversine formula. Distance *d* is a function of two latitude and longitude coordinates (φ_1, γ_1) and (φ_2, γ_2). Where *r* is the radius of the Earth.

$$d(\varphi_1\, \gamma_1\, \varphi_2\, \gamma_2) = 2r\ \ arcsin\left(\sqrt{sin^2\left(\frac{\varphi_2 - \varphi_1}{2}\right) + \cos(\varphi_1)\, cos(\varphi_2) sin^2\left(\frac{\gamma_2 - \gamma_1}{2}\right)}\right)$$

The raw GPS data distance calculation method was selected because it is a good baseline of comparison for the other methods. The raw GPS data distance calculation does not factor in the natural error of GPS readings discussed in section 1.2. It would be expected that the raw GPS data method will over estimate the total distance traveled. This can be illustrated by considering the distance traveled between two points A and B. One path is a straight line between A and B and the other path is a zig-zag line between A and B. If the total distance of each path is calculated, the zig-zag path will be longer than the straight-line path. To relate this back to the raw GPS data distance calculation method. The straight line would be the actual path taken and the zig-zag path would be the path calculated due to the error in the GPS readings. See Algorithm 1 for pseudocode of the raw GPS data distance calculation.

Algorithm 1 Pseudocode for distance calculation using raw GPS data. This method is called every time the system receives a GPS update from the system. The *distance*, *prev_lat*, and *prev_lon* variables are persistent between calls to this method. The *haversine* function is defined by Formula 1.

```
UpdateDistance (lat, lon):
   if prev_lat != null && prev_lon != null:
      distance += haversine(prev_lat, prev_lon, lat, lon)
   prev_lat = lat
   prev_lon = lon
```

2.2 GPS Speed Data

The next method evaluated was calculating distance from speed. The Android Location API (Google, 2018) exposes a speed value along with the latitude and longitude coordinate for each location update. The speed value attempts to capture the magnitude of the velocity the device is moving at the time of the location update. The speed value is calculated by the GPS receiver by calculating the Doppler shift of the satellite signals received. The Doppler shift can be calculated by comparing the frequency received from each satellite against the expected frequency. With the calculated Doppler shift value, a velocity can be determined. (Hofmann-Wellenhof, et al., 2012, p. 6). Given that only the total distance of an activity is calculated and not the path taken, the velocity value alone can be used to calculate distance between two points. The distance traveled since the last location updated is calculated by dividing the speed at the current update by the time that has passed since the previous update. See Algorithm 2 for pseudocode of the GPS speed distance calculation.

This method was chosen because it does not rely on the actual GPS coordinates provided by the receiver. It is possible that the speed reading has a smaller margin of error than GPS coordinate readings. If this is the case the error of GPS location data is a non-issue because the locations will not be used. However, the results show that at low speed, the speed reading is highly inaccurate.

Algorithm 2 Pseudocode for distance calculation using GPS speed data. This method is called every time the system receives a GPS update from the system. The *distance* and *prev_time* variables are persistent between calls to this method.

```
UpdateDistance (speed):
   if prev_time != null:
      distance += speed / (Time.Now - prev_time)
   prev_time = Time.Now
```

2.3 Averaging GPS Data

The next method evaluated was calculating distance by averaging a set of raw GPS coordinates. This method works by collecting a set of *n* coordinates and averaging them to form a single point. Next, another set of *n* coordinates are collected and averaged. The distance between the two averaged locations is added to the overall distance. The distance between the two coordinates is calculated using Formula 1. See Algorithm 3 for pseudocode of the GPS averaging distance calculation. Although this algorithm is similar to the algorithm in Eliasson, 2014, it was not derived from that work.

This method was chosen because it is more resilient to the error of GPS coordinate readings. The assumption is that the error is distributed evenly among a set of GPS readings in relation to the actual location of the device. Given this assumption it can be concluded that the average of a set of points will be closer to the actual location than an individual reading. However, there is a tradeoff between having a large and small *n* value. Larger *n* values will reveal a coordinate closer to the actual location but will have a longer delay in calculating the distance value due to the overhead of collecting all *n* coordinates before calculating the distance. Due to this delay, the distance calculated may cut out large portions of actual distance traveled if the actual path taken is not in a straight line. Smaller *n* values will result in a less accurate coordinate in relation to the actual location of the receiver as compared to larger *n* values, but will calculate the distance value more frequently. The higher frequency calculation will result in less smoothing of curved paths.

Algorithm 3 Pseudocode for distance calculation using the average location of a set of GPS coordinates. This method is called every time the system receives a GPS update from the system. The *distance*, *prev_lat*, *prev_lon*, and *coord_list* variables are persistent between calls to this method. The *haversine* function is defined by Formula 1.

```
UpdateDistance (lat, lon, set_size):
   coord_list.Add(vector(lat, lon))

   if coord_list.Size == set_size:
      avg_lat = 0
      avg_lon = 0

      for point in coord_list:
         avg_lat += point.x
         avg_lon += point.y
      avg_lat /= set_size
      avg_lon /= set_size

      if prev_lat != null && prev_lon != null:
         distance += haversine(prev_lat, prev_lon, avg_lat, avg_lon)

      prev_lat = avg_lat
      prev_lon = avg_lon
      coord_list.Clear()
```

2.4 Line of Best Fit

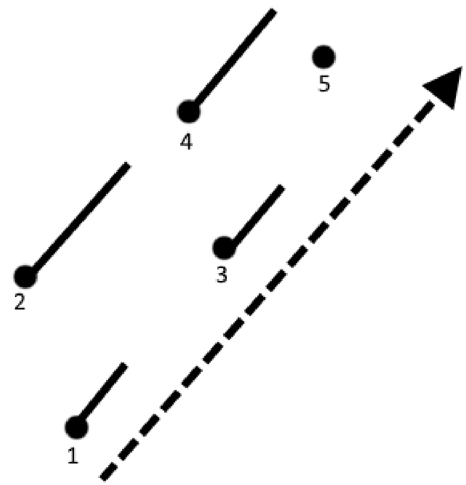
The final method evaluated was calculating distance by looking at the line of best fit for a set of points. This method works by collecting a set of *n* coordinates and calculating the line of best fit for those coordinates. After that the estimated length traveled along that line is calculated and added to the overall distance. The line of best fit is calculated using the least square method. The slope of this line is turned into a normalized directional vector. Using the normalized directional vector of the line of best fit, a scalar project is performed with the vector between two consecutive points to determine the length traveled along the line of best fit. The Haversine formula from Formula 1 is used to calculate the distance along the line segment. The sum of these distances calculates the total distance traveled for that set of *n* coordinates. See Figure 1 for a visualization of the calculation. See Algorithm 4 for pseudocode of the GPS averaging distance calculation.

This method was chosen because, like the averaging method, it is more resilient to GPS error. The difference between this method and the averaging method is that this method attempts to estimate the actual path taken by the receiver rather than estimating the position of the receiver. Like the averaging method there is a tradeoff for using a larger and smaller *n* value. Larger *n* values will result in a more accurate line of best fit for activity moving in a relatively straight line. However, if the path of the activity is along a curved path, the line of best fit will smooth the calculated path taken, resulting in a smaller distance calculation than what occurred in reality. Smaller *n* values will result in a less accurate line of best fit but due to the more frequent updates will not smooth the calculated path taken as much as larger *n* values.

Algorithm 4 Pseudocode for distance calculation using the line of best fit of a set of GPS coordinates. This method is called every time the system receives a GPS update from the system. The *distance* and *coord_list* variables are persistent between calls to this method. The *haversine* function is defined by Formula 1. The *dot* function is a vector dot product.

```
UpdateDistance (lat, lon, set_size)
   coord_list.add(vector(lat, lon))

   if coord_list.size == set_size:
      avg = vector(0,0)

      for point in coord_list:
         avg += point
      avg /= set_size

      rise = 0
      run = 0

      for point in coord_list:
         rise += (point.x – avg.x) * (point.y – avg.y)
         run += (point.x – avg.x) ** 2

      best_fit = vector(rise, run).normalized
      last_point = coord_list[0]

      for i = 1 to set_size:
         // Vector projection
         p = dot(best_fit, coord_list[i]-last_point)
         // Distance along the project line segment
         distance += haversize(last_point, last_point + (best_fit*p))
         last_point = coord_list[i]

      coord_list.clear()
      // The last point is added to the list to not lose distance between sets
      // of points
      coord_list.add(last_point)
```

Figure 1 Visualization of the line of best fit distance calculation method. The numbers indicate order the points were received. Note that the points are GPS coordinate readings not the actual location of the receiver. The dashed line shows the directional vector of the line of best fit. The solid lines show the distance calculated.



CHAPTER III

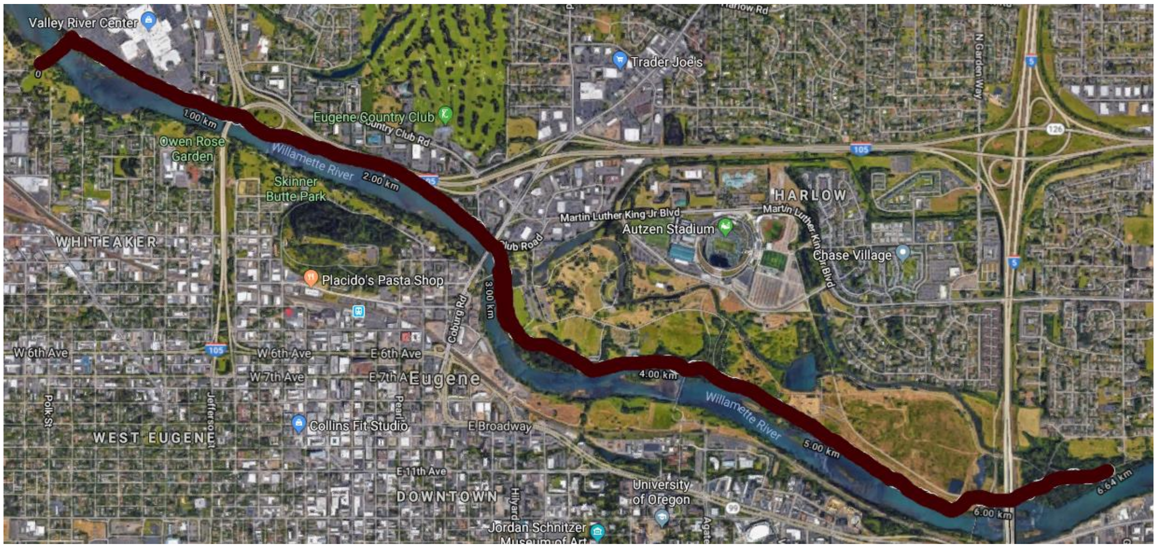
EXPERIMENTAL TESTING

Each method described in Chapter II was implemented at tested on a variety of activities. For both the averaging method and line of best fit method a *n* value of 5 and 10 were tested. The three activities chosen to test were; biking, walking, and idle. The GPS data was collected using BLU Advance A4 Android devices. Each device was running Android operating system version 6.0. Six devices were used in total. The GPS service on each device was configured to receive a location update approximately one time every second. For each activity tested, four trials were conducted. All six devices were set to collect GPS data for each trial resulting in 24 data points per activity. The following pieces of data were collected on each device per trial: total distance calculated for each distance calculation method, average speed of the device, and maximum speed of the device. All GPS updates for each trial were saved to the device to enable playback of the GPS data at a later time. All trials were conducted on days with clear skies. Optimal weather conditions for GPS data collection.

3.1 Biking

Biking was chosen because it is one of the highest speed non-motor assisted activities commonly performed. The biking trials were performed on a biking trail rather than a street to ensure consistent trials. To simulate biking on the street where stopping at intersections is common, there was a 30 second stop for every mile of riding. See Figure 2 for an approximate mapping of the path taken. Two trials were riding East to West and two trials were riding West to East. The exact same path was followed for all four trials. The total actual distance of each trial was 4.12 miles.

Figure 2 Approximate path taken (in red) for the biking trials mapped via Google Maps



3.2 Walking

Walking was chosen because it is very low speed and often follows a path that has abrupt changes in direction. It is important that the distance calculation is accurate for both high speed and low speed activity. The walking trials were performed on a residential street. See Figure 3 for an approximate mapping of the path taken. All four trials followed an identical path. The total actual distance of each trial was 1.0 miles. Each trial was performed continuously without any stops except for a brief stop at the end to stop recording the GPS data on each device.

3.3 Idle

An idle test was performed to test the accuracy of the distance calculation with no movement. Many times, activity such a biking or walking will have brief moments of stopping, stop lights, cross walks, etc. It would harm the accuracy of the overall distance calculation to have these moments of stoppage increased the overall distance calculated. Given the natural error of GPS readings it would be expected that many of the methods tested would erroneously calculate non- zero distance while the device is stationary. The idle test was performed by placing the devices on a table outdoors and collecting GPS data for 10 minutes.

Figure 3 Approximate path taken (in red) for the walking trials mapped via Google Maps

3.4 Results

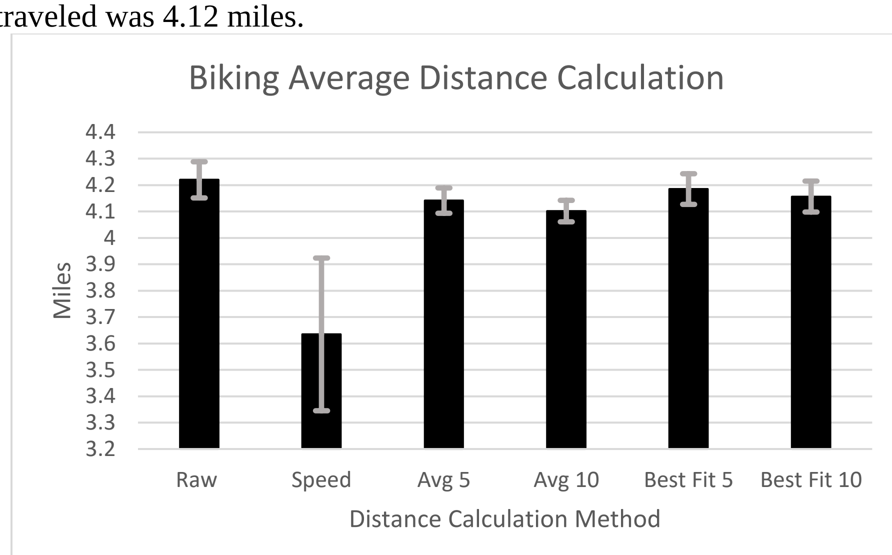
3.4.1 Biking

See Figure 4 for the distance calculation results for the biking trials. The biking trials resulted in an overall average speed of 4.409 meters per second and an average maximum speed of 8.166 meters per second. The results show that the averaging method with an *n* value of both 5 and 10 are extremely accurate in the distance calculation with low variance. The line of best fit and raw GPS data methods both tend to overestimate the total distance but also have low variance. The speed method on average vastly underestimates the total distance and has a high variance.

3.4.2 Walking

See Figure 5 for the distance calculation results for the walking trials. The walking trials resulted in an average speed of 1.483 meters per second and an average maximum speed of 5.826 meters per second. The average maximum speed of 5.826 meters per second (13 miles per hour) is not a realistic walking speed. The most likely reason for this high maximum speed is due to the walking path being on a residential street. Being next to buildings or houses can have a great effect on the accuracy of the GPS reading. The results show that the raw GPS data method is the most accurate with low variance. The averaging and line of best fit methods both underestimate the total distance traveled with low variance. The line of best fit method with both *n* value of 5 and 10 is slightly more accurate than the averaging methods. Like the biking trials, the speed value vastly underestimates the total distance with a high variance.

Figure 4 Results of the biking distance calculations. The bars in black represent the average distance calculated. The gray bars represent the standard deviation. Actual distance traveled was 4.12 miles.



3.4.3 Comparison of Biking and Walking

To easily have a visual comparison of the methods the average percent error of each method was calculated, see Figure 6. Based on this comparison it is clear that the speed method does a very poor job at estimating the distance traveled for each activity. There is a large difference in error with both the averaging and line of best fit methods between biking and walking. The most likely explanation of this must do with the type of paths taken for each of the trials. The biking trials followed a relatively straight path while the walking trials followed a path that had many curves and corners. These corners were most likely cut off by the smoothing of the averaging and line of best fit methods resulting is an underestimate of the total distance. For the raw GPS method there was very little difference in the percentage of error between walking and biking. This further supports the idea that the smoothing of the averaging and line of best fit methods result in an underestimate of distance for winding paths.

Figure 5 Results of the walking distance calculation. The bars in black represent the average distance calculated. The gray bars represent the standard deviation.

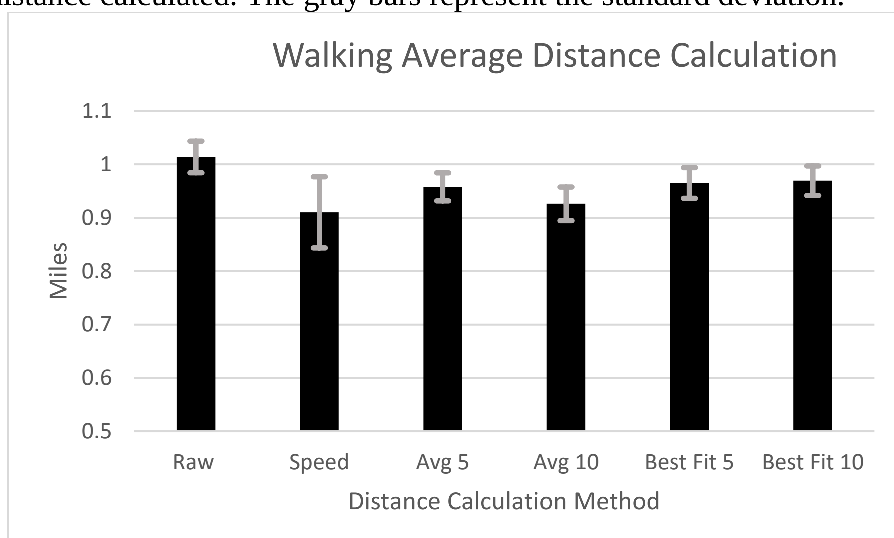
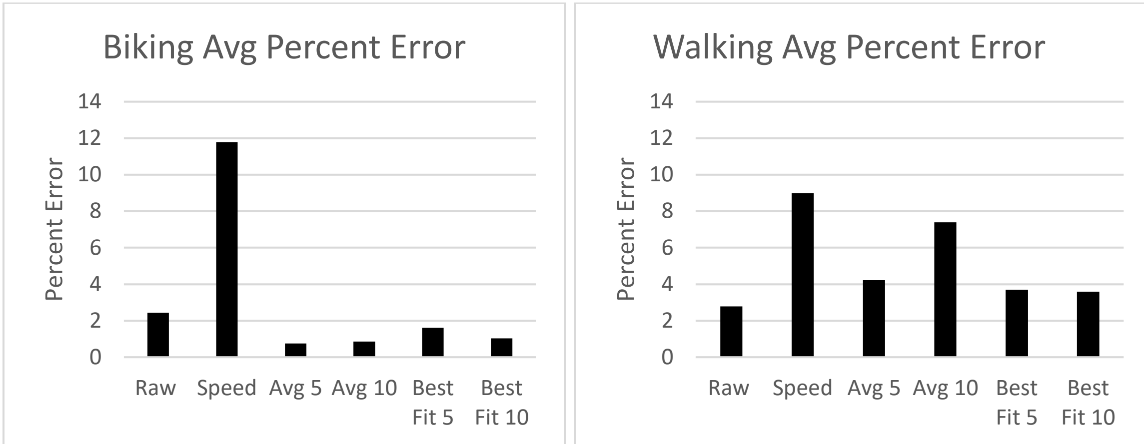


Figure 6 Percent error of the total distance traveled of each method for (a) biking and (b) walking

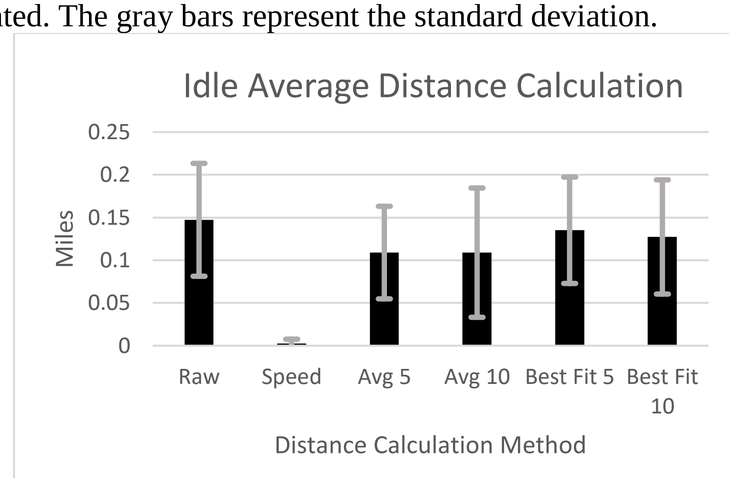


(a) (b)

3.4.4 Idle

See Figure 7 for the distance calculation results for the idle trials. The idle trials resulted in an average speed of 0.0065 meters per second and an average maximum speed of 0.586 meters per second. As expected, the methods that rely on GPS coordinates generated a significant amount of distance. This is due to the natural error in the GPS readings from the device. More interestingly, the speed method is the only accurate method for calculating distance while not moving. While the speed may be highly inaccurate at low speed, it is highly accurate at no speed.

Figure 7 Results of the idle distance calculation. The bars in black represent the average distance calculated. The gray bars represent the standard deviation.



CHAPTER IV

GPS DATA FILTER

Looking at the results from Chapter 3, there is no method that accurately calculates distance for all three activities; biking, walking, and idle. The raw GPS data, averaging, and line of best fit methods do a reasonably good job calculating the distance while in motion but do a poor job while idle. On the other hand, the speed method does a poor job calculating distance while in motion but does a great job calculating distance while idle. Given this dichotomy, a filter can be implemented that retains the accurate distance calculation of the non-speed methods while in motion while filtering out data while idle. The calculation methods will remain the same as they are explained in Chapter 2, but if the speed reading is below a certain threshold the GPS update will be filtered out. The threshold value was determined by looking at the average maximum speed value calculated from the idle trials from section 3.4.4. The threshold value used for the GPS data filter was 0.6 meters per second. If any speed is below 0.6 meters per second the update will not be considered in the overall distance calculation.

4.1 GPS Data Filter Results

The distance calculation with the GPS data filter was evaluated on the same data that was collected from the original experimentation using the GPS data playback explained in Chapter 3. See Figure 9 for the distance calculation results with the GPS data filter. The most important result is that all distance calculation methods had nearly zero distance calculated for the idle test. To better view how the distance calculation changed with and without the GPS data filter the percent improvement was calculated for each distance calculation method with the filter without the filter. See Figure 8.

The filter showed great improvement for the idle trials for every method except the speed method. The reason the speed method saw such a large decrease in accuracy is due to how the distance is calculated for that method. The speed method takes the time between consecutive updates multiplied by the speed of the update. Because the filter is throwing out data that is below the speed threshold, whenever the device is idle, most of the readings are being filtered out. This means there will be a relatively large span of time between consecutive updates causing the speed to calculate more distance when an update occurs that is not filtered out. Given the poor performance of the speed method while in motion this is not an issue.

The averaging method saw a decrease in accuracy for both biking and walking. This implies that the averaging method underestimates the overall distance. For the biking trial, the reason the averaging method without the filter was so accurate was because the distance while stopped was being added to the overall distance. Had those stops been longer the averaging method would begin to overestimate the distance. With the filter this error was filtered out leaving the total distance calculated shorter than without the filter, resulting in a decrease in accuracy.

The line of best fit method saw in increase in accuracy for biking but a decrease in accuracy for walking. The line of best fit method with both a n value of 5 and 10 overestimated the distance without the filter. Filtering out the error generated by the stops resulted in a distance extremely close to the actual distance traveled. The line of best fit method with a n value of 5 had an average error of 0.0256 miles for a 4.12 mile route. However, there was a significant decrease in accuracy in the walking trials. The most likely cause of this is that the data would be filtered out more often than it did relative to biking. Given that walking is a much slower activity than biking and the relatively large standard deviation of the speed data. There will be times the data is filtered out due to error in the speed reading even though the actual speed is above the threshold. This extra filtering will cause more smoothing in the estimated path calculated by the line of best fit. Resulting in an underestimate of the total distance of that path segment.

The raw GPS data method saw an increase in accuracy for both the biking and walking trials. The raw GPS data method does not suffer from the smoothing effects that the averaging and line of best fit methods do. The filtering for the biking trials filtered out the updates while stopped. This decreased the total distance traveled. Without the filter this method overestimated the overall distance traveled. With the filter the overall distance calculation still overestimated, but not by as much. Because the raw GPS data method does not suffer from the smoothing effects like the averaging and line of best fit methods do, the walking distance calculation is highly accurate. With an average error of 0.024 miles for a 1 mile route.

Figure 8 Percent improvement of the accuracy of the distance calculation with the filter over the distance calculation without the filter.

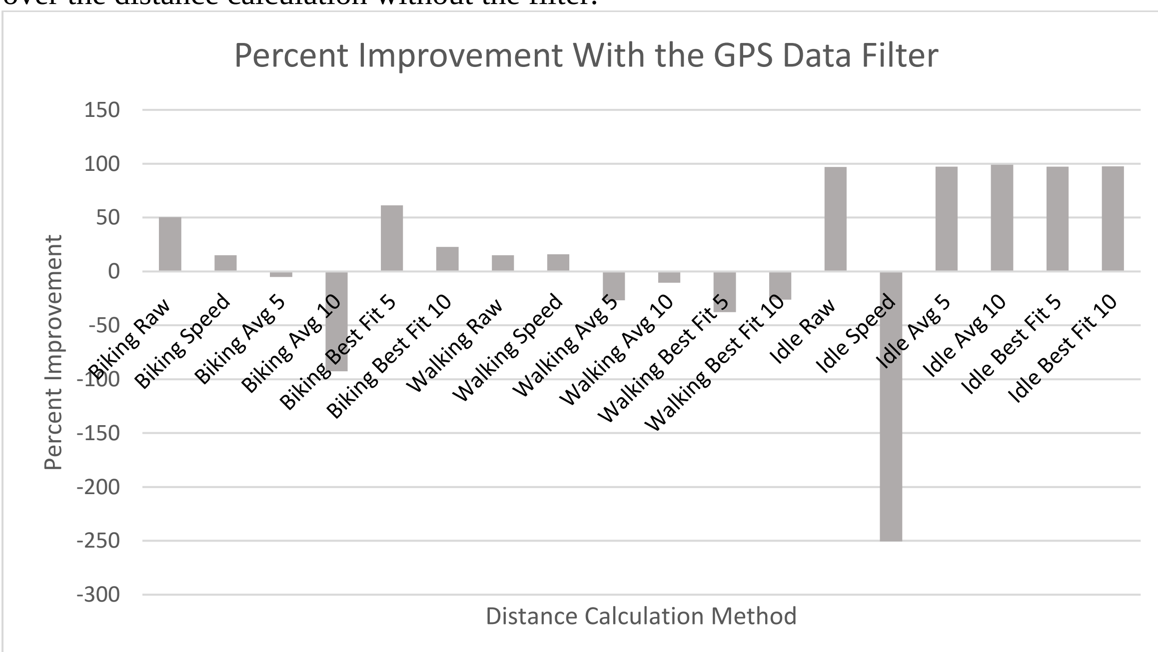
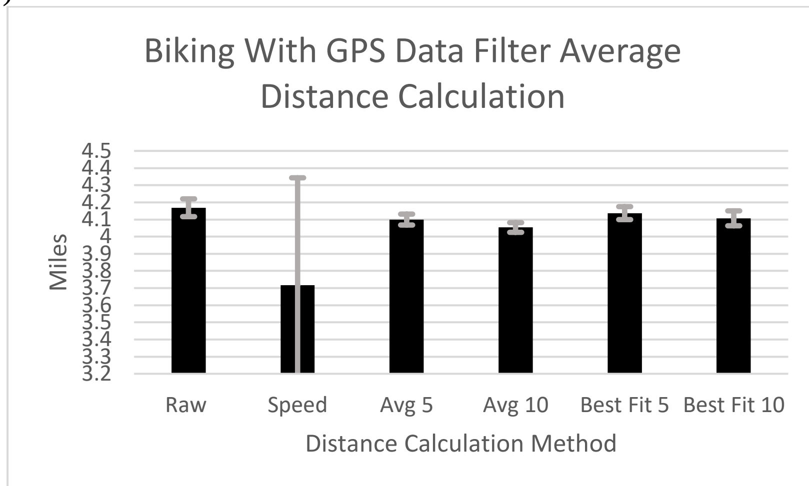
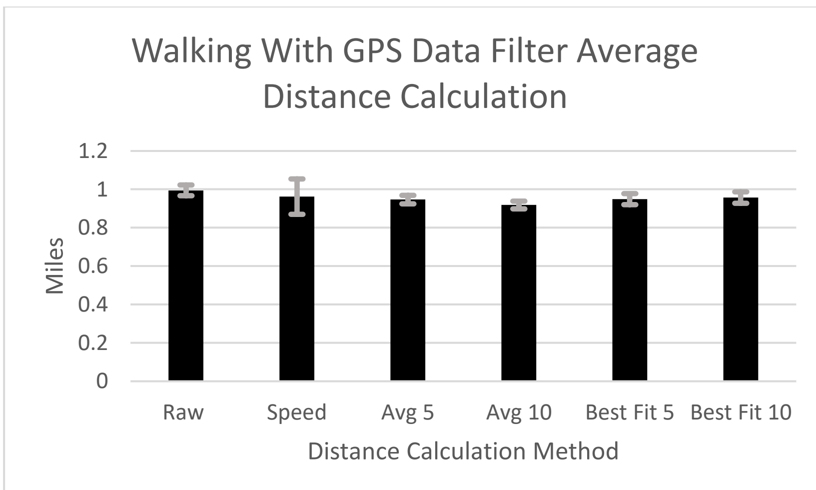


Figure 9 Results of the distance calculation with the GPS data filter for (a) biking, (b) walking, and (c) idle.

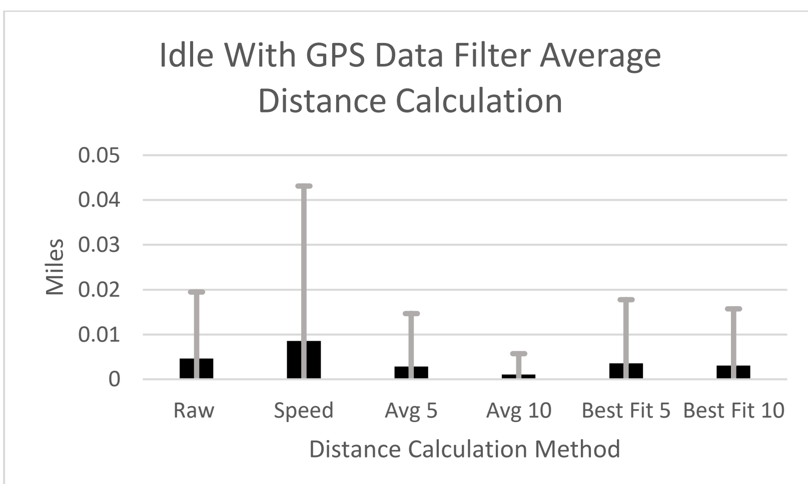
(a)



(b)



(c)



CHAPTER V

EXAMPLE USE CASE: POCKET BIKE

While calculating the total distance of an activity is interesting, it does not provide much value unless it is being used to achieve a goal. It provides a framework on which applications can be built. The Eugene Springfield Safe Routes to School (SRTS) group identified the need for a product to help promote physical activity for children. There has been previous research in this area looking at ways to reward children who either walk or bike to school (Yang, Harbaugh, & McDonald, 2015). This worked by having kids carry a RFID tag with them and when they either walked or biked to school they would scan the tag. They could then earn rewards based on how many times their tag was scanned. One of the limitations of this study is that it does not consider the proximity of the children to the school. The distribution of effort vs reward is not even for all students. A student who live close to school need to put in much less effort to get the same reward as a student who lives far from school. In some cases, a student might live so far from school where biking or walking would not be a possibility making it impossible for that student to participate altogether.

To address these issues an application was created that used the distance of an activity as a basis for the reward. This application is called Pocket Bike. Pocket Bike is a video game developed to work on any Android device that has GPS capabilities. The video game itself is a race. The player controls a bike and the goal is to get to the finish line as fast as possible without crashing. There are many hills and dips on the course that make the game challenging. The top times for each level are displayed on a leaderboard. The leaderboard is used as the primary motivational tool to keep the player engaged. It creates a competitive aspect to the game and gives the player a goal to achieve.

The GPS distance tracking is used to generate points which are required to play the game. The line of best fit method with a *n* value of 5 was used for distance tracking due to the high amount of accuracy for biking. The app tracks the total distance traveled and rewards the player 100 points per mile of activity; 100 points are needed to unlock a level for 12 hours and there are three levels total. Using points to unlock game levels is a variation of the microtransaction business model used in many mobile games today. The microtransaction model works by letting the user play the game for free but offering low-cost items to purchase in the game with real money. Pocket Bike follows this same model but instead of using real money to purchase items, the player will use activity points. See Appendix A for the user-guide for Pocket Bike.

5.1 Evaluation

To evaluate the app, Pocket Bike was given to members of the SRTS team as well as select faculty members at the University of Oregon. Each evaluator was asked to use the app for a few days and answer the following questions: (1) “Overall what are your thoughts on the app as a whole?”, (2) “Overall how accurate did you find the distance tracking?”, (3) “How effective do you think this app would be at motivating kids to become more physically active?”, (4) “What could be changed to added to this app to improve the effectiveness of motivating kids to become more physically active?”, and (5) “Can you thank of any other applications (besides games) that could use mobile distance tracking to achieve the goal of promoting physical activity for kids?”.

The evaluators found the distance tracking to accurately reflect the distance they traveled. There was one exception, during one bike ride an evaluator found the distance to be about a half mile shorter than they expected it to be. Upon further discussion the reason for the discrepancy was determined. Because Pocket Bike rewards physical activity there was an upper speed limit introduced to the data filter to avoid point collection while riding in a car. If the device was traveling over 23 miles per hour for a continuous 10 seconds it would ignore all GPS updates for 1 minute. The evaluator had a portion of their bike ride that was downhill. This caused them to go over 23 miles per hour and triggered the distance calculation to ignore GPS updates. This accounted for the half mile of distance missing from the total calculation. Because Pocket Bike is targeted toward children it would be unlikely this situation would occur often. Although it would be beneficial to investigate alternative methods of detecting the type of activity the user is performing. The type of activity being performed would be the determining factor on whether or not GPS updates are ignored.

Overall, none of the evaluators had a strong opinion on Pocket Bike itself but they did provide insight on other uses of distance tracking to promote physical activity. The best example of this is the idea to use tracked activity as a requirement to unlock the device. The user would be required to travel a specified distance in order to gain access to the device. With Pocket Bike, if the user does not find the game fun they will have no incentive to go out and do physical activity to play the game. Using distance to unlock the device would avoid that problem by making “the game” any use of the device. Every child might have a different activity they enjoy using the device for such as; watching videos, listening to music, playing games, etc. This idea would provide a high level of motivation for that child to become my physically active.

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

In this thesis it was shown that using GPS we can effectively calculate distance traveled of low speed activities. All the methods tested, except for the speed method, yielded consistent results that were all acceptably close to the actual distance traveled with some methods being slightly more accurate than others. While the speed was not effective at calculating distance, it was effective at providing a means to filter out data while not in motion. However, it was not found that a single method is clearly the best at calculating distance for all types of activity. Biking saw the line of best fit method with a n value of 5 being the most accurate method while walking saw the raw GPS method as being the most accurate. There are many factors that play into which method will be best for a given situation. These factors are things such as the average speed of the activity, the typical type of path taken (straight lines or winding paths), and the type of surrounding in the area the activity is typically perform in. If the activity will occur near buildings this can have an impact on the overall distance calculation. Ultimately it comes down to choosing the method that will best suited for the expected use cases of the application.

6.2 Future Work

More work can be done to test alternative methods of calculating distance, such as a Kalman filter. In the latest version of the Android API, the Location Services module makes available many more estimations of error as calculated by the GPS receiver. The most useful of these error calculations would be the estimated error for the speed and bearing values (Google, 2018). These error values could be used in a Kalman filter to better estimate the actual location of the GPS coordinate.

Another direction that could be considered is changing the distance calculation methods based on what activity is being performed. The results show that different calculation methods produce more accurate results depending on the activity being done. The accelerometer could be used to predict what activity is currently being performed and adjust the distance calculation method to choose the most accurate version for that activity as explained in Ravi, Dandekar, Mysore, & Littman, 2005. This would also remove the need for the speed filter which showed to filter out a little too much data while walking as the accelerometer would be able to detect when the device is idle. It would also give a better criterion to be used to filter out activity that should not count toward the overall distance calculation. This could be used to avoid the issue discussed in section 5.1.

There also needs to be work analyzing what happens to the GPS data while in locations that are not optimal for GPS, such as being inside a building or near a building. An observation made while testing the GPS distance calculation was that while indoors the device would often not have any GPS signal and when the device did get a signal it would quickly generate distance while being idle due to the extremely large amount of error. This problem would need to be solved in order to have GPS activity tracking that is constantly running in the background.

APPENDIX

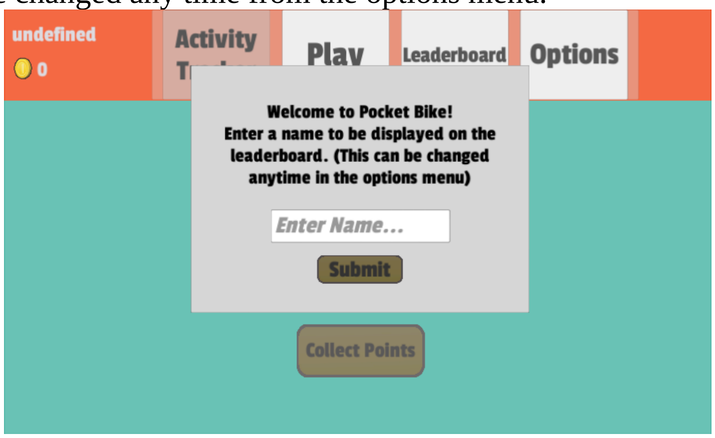
POCKET BIKE USER GUIDE

Pocket Bike User Guide

5/22/18

First Time Startup

Upon launching the app for the first time you will be prompted to input a name. This is the name that will be used on the leaderboard. After providing your name click “Submit”. This name can be changed any time from the options menu.



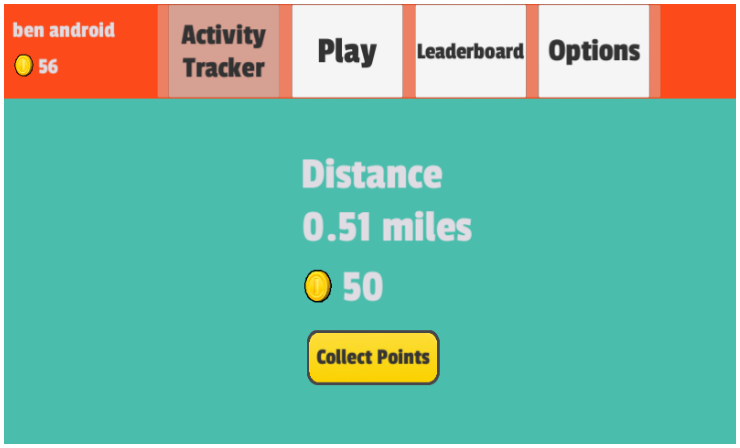
Activity Tracker

Overview

Navigate to the “Activity Tracker” menu item to view the Activity Tracker information. Activity is how you generate points to purchase level unlocks. The Activity Tracker tracks the distance you travel while performing physical activity (biking, running, walking, etc.) The Activity Tracker becomes active as soon as you launch the app and will continuously track distance while the app is running. The Activity Tracker remains active the entire time the app is running. This includes while the app is running in the background. To disable the Activity Tracker the app must be killed. If you kill the app with points you have not collected, those points will still be there to be collected when you re-launch the app.
Note: The Activity Tracker does not store the GPS data collected. GPS data is only used it to calculate the distance traveled.

Collecting Points

While performing an activity the activity tracker will display the distance measured along with the Points generated from the activity. The Points are identified by the <points img> symbol.



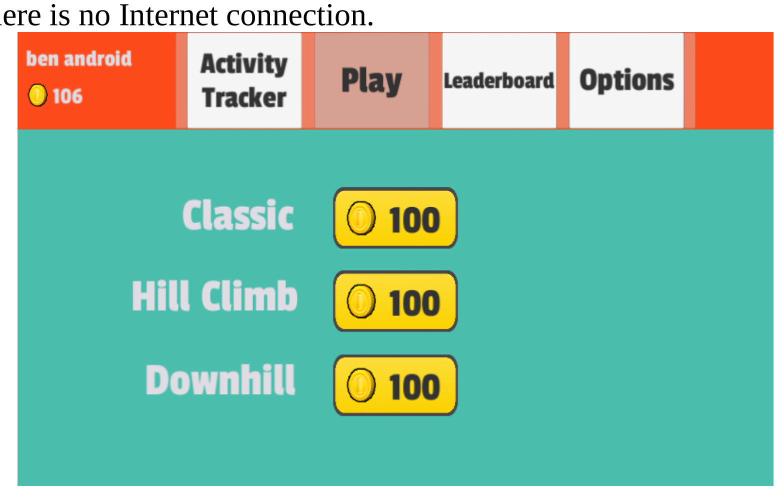
To collect the Points generated to use to purchase levels the "Collect Points" button must be pressed. Upon successful collection the Points will be added to the Point total in the upper left portion of the screen.

Note: To collect Points there must be an active Internet connection. An error will be displayed if there is no Internet connection.

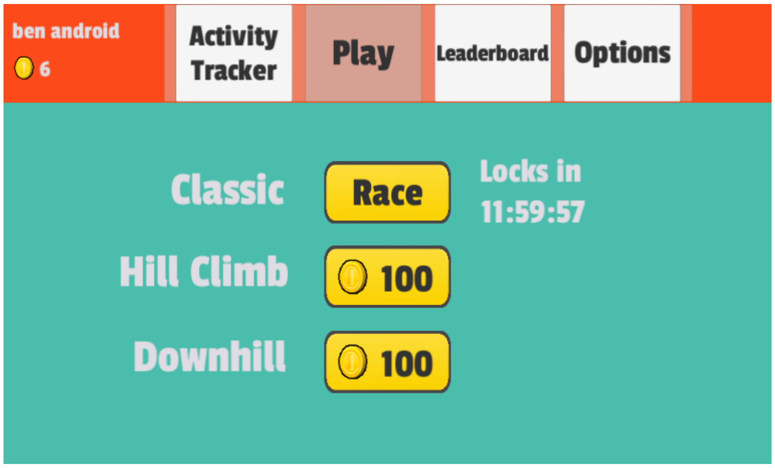
Play Menu

In the Play menu there will be three levels listed; Classic, Hill Climb, and Downhill. To play a level, you must first generate and collect 100 Points. Once the points have been collected, click the button next to the level you wish to purchase. If you do not have enough points to purchase the level an error will be displayed.

Note: To purchase a level you must have an active Internet connection. An error will be displayed if there is no Internet connection.

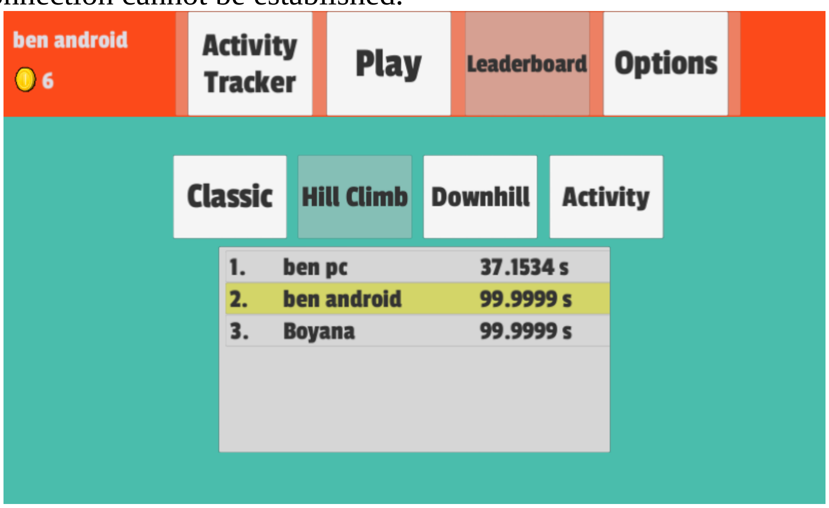


After a successful purchase of a level, that level will remain unlocked for 12 hours. A countdown timer next to the level tells you how much time is left. After 12 hours the level becomes locked and must be unlocked with another 100 points. When a level is unlocked click to "Race" button to play that level.



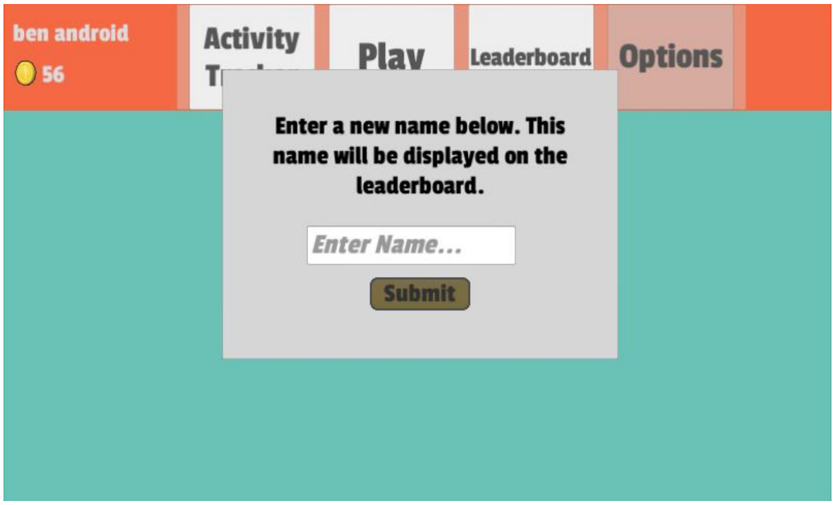
Leaderboard Menu

Navigate to the Leaderboard menu option to view the global leaderboard. In the Leaderboard menu click the button for the level you wish to view the leaderboard for. There is a leaderboard for each of the three levels along with the total distance traveled by the Activity Tracker. Your location on the Leaderboard will be highlighted in yellow. Note: Internet connection is required to view the Leaderboard. An error will be displayed if Internet connection cannot be established.



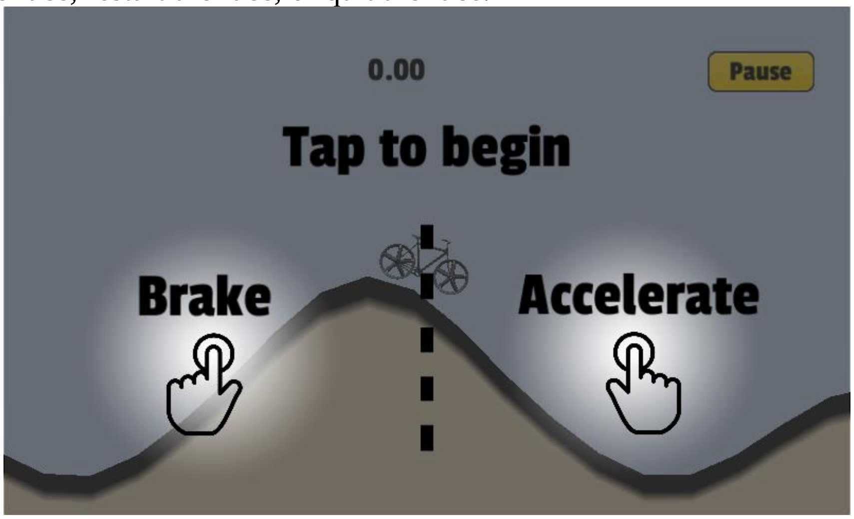
Options Menu

Navigate to the Options menu option to view the options for the app. Currently the only option available is to change the name displayed on the leaderboard. To change your name in the app, click the “Change Name” button. A window will be displayed prompting for a new name. After entering a new name click the “Submit” button.



Playing the Game

To play the game you first must purchase a level unlock. When you click the "Race" button in the Play menu you will be taken to a race.
To begin a race, tap anywhere on the screen. The goal of the race to is make it to the finish line as fast as possible without crashing. The controls for the race are as follows: tap the left side of the screen to brake and flip forward, tap the right side of the screen to accelerate and flip backward. You can pause the race at any time by clicking the "Pause" button in the upper right portion of the screen. The pause menu gives you the options to continue the race, restart the race, or quit the race.



Upon completion of a race or a crash you are given the options to "Play Again", or "Go Back". "Play Again" will restart the race and "Go Back" will bring you back to the main menu.

REFERENCES CITED

Cho, D. K., Mun, M., Lee, U., Kaiser, W. J., & Gerla, M. (2010, March). Autogait: A mobile platform that accurately estimates the distance walked. In *Pervasive computing and communications (PerCom), 2010 IEEE international conference on* (pp. 116-124). IEEE.

Chopde, N. R., & Nichat, M. (2013). Landmark based shortest path detection by using A* and Haversine formula. *International Journal of Innovative Research in Computer and Communication Engineering*, *1*(2), 298-302.

Eliasson, M. (2014). A Kalman filter approach to reduce position error for pedestrian applications in areas of bad GPS reception.

Godha, S., Lachapelle, G., & Cannon, M. E. (2006, September). Integrated GPS/INS system for pedestrian navigation in a signal degraded environment. In *ION GNSS*(Vol. 2006).

Google. (2018, June). Android Documentation – Location. Retrieved from https://developer.android.com/reference/android/location/Location.

Grewal, M. S., Weill, L. R., & Andrews, A. P. (2007). Global positioning systems, inertial navigation, and integration.

Hofmann-Wellenhof, B., Lichtenegger, H., & Collins, J. (2012). Global positioning system: theory and practice.

Ravi, N., Dandekar, N., Mysore, P., & Littman, M. L. (2005, July). Activity recognition from accelerometer data. In *Aaai*(Vol. 5, No. 2005, pp. 1541-1546).

Yang, Y., Harbaugh, W., & McDonald, N. (2015). *Encouraging Active School Travel by Making it" Cool": A Quasi-Experimental Study Using Boltage* (No. NITC-RR-550).