Web-based Singletrack Trail Information System & Geolocator

SIE 509 – Fall 2015: Final Project Report

Joel Whitney (joel.whitney@maine.edu)

Introduction

When I first starting thinking about projects for this class I wanted to come up with an idea that was both novel and immediately useful for myself and others. I was biking a lot of the singletrack mountain bike trails in the Bangor area, but I wasn't very familiar with the Orono/Old Town trail network. I decided to start bringing my bike with me when I came to campus so I could explore the trails after classes. The first time I went out for a ride, I remember I ended up getting turned around and lost numerous times. On my second trip, I ended up coming out of the woods at the mill in Old Town and biked pavement back. I thought to myself, if I can get this turned around on a relatively small trail network than it must be extremely hard to navigate through larger trail systems. Since I, and most other bikers, carry mobile devices on their person it makes sense to have a system that not only aids in traversing trail systems, but also can provide trail information to its users.

The traditional solution for users of trail systems is to reference a trail map of the geographic area. Because of this, most management groups of trail systems have some sort of map that trail users can use to see the layout of the trails. A lot of the times this is in the form of a paper map at a kiosk or on signage at the trail entrances. Sometimes you can find PDF versions of these maps available online for use. Very rarely, albeit becoming more frequent nowadays, will you see PDFs with the georeferencing maintained so that the PDFs can be used for navigation. Even in these rare cases, the user is still unable to directly interact with the map to acquire additional information about the trails besides location. Furthermore, for bikers who visit multiple trail systems, keeping track of multiple PDF maps is a major inconvenience. Based on the above issues, it was clear to me that a better system should exist for managing location and trail system information.

My plan early on was to make a central database of all of the trail networks that have single track bike trails in the area. I decided to start with the Orono trail systems first and then work on the Bangor systems if time permits. This forced me to consider making a system that could be easily extendable to other systems. When using the system I wanted the users to be able to view numerous attributes about the trails, such as: local trail names, IMBA trail ratings, land management, pets allowed, etc. I wanted to take this database and make it available on an easy to use mobile web map that automatically fixes to the persons current position using the phones built in GPS. Initially, from my limited experiences with web maps, I thought this would result in a HTML page with a Leaflet based map that allows users to click the trail and have a popup generated with the appropriate info.

Methods

From my initial research, it looked like there were a handful of PDF maps available for the trails in the area. Most of the maps were a number of years old and the data was not easily accessible. Starting with the Orono system, I emailed the Forestry Department and was able to acquire the trail data used to create the map for the DeMeritt Forest. The data that was used to create the trail maps was from 2009 and was incomplete, especially for the singletrack trails. I was going to have to collect my own data to get a complete dataset of the DeMeritt Forest.

Initially, I started by using Strava, my biking app, to record rides that I went on. After a few attempts, I realized that the combination of collecting data on my mobile GPS under tree cover made it extremely

difficult to get consistent data. The general flow of the trails was consistent and I figured the more data I could collect the easier it would be to build centerlines from the data. Ultimately, I would use multiple modes of collection to try and get the most data to work with as possible. This would mean using the data I already had from the University as well as collecting data through: the Strava app on my mobile phone, my Garmin 60cx handheld GPS, and an open-source data collection app called GeoODK Collect.

Collecting the data using the Strava biking app and the Garmin 60cx was essentially the same process. When I departed from the Recreation Center parking lot I would turn on my GPS and the start the Strava app on my phone. I would also start the tracking on my Garmin handheld. At the end of the ride I would stop the Strava app and save the ride as an easily identify track, such as 'UMOTrailSearch_10_02_15'. I would also stop the tracking on my Garmin and save the track so the SD card. I was then able to go online on my Strava account and download the GPX file for each biking session. For my Garmin handheld, I just had to mount the SD card on my computer and I could extract the GPX from each session. In total, I biked 51.8 miles collecting data using the Strava app. For some reason, ArcMap did not like the GPX files from my two sources so I imported them into QGIS first and then exported them as shapefiles that I could then add into my geodatabase.

In addition to having accurate paths for each trail, I also wanted to have various trail information for the trails, such as: local trail names, IMBA trail ratings, land management, pets allowed, etc. This meant I would need to have some sort of data collection method. I decided to utilize an open source data collection app, called GeoODK Collect, which I had prior experience with. GeoODK Collect uses XML based forms that the user can fill out on their mobile device and send back to an aggregate server through an internet connection. This allowed me to collect trail paths and information in the woods and send it back to the server in real time. I spent some time and programmed two forms that would be used by the app for my data collection: 1) 'Trail_Info_CollectorV2.xml' and 2) 'Feature_Collector.xml'. The trail info form allowed the user to input data such as: trail name, record the trail path (at 1 sec intervals), trail type, various IMBA rating categories, and add additional comments if desired. The feature collector form let the user input location (GPS), feature type, take a picture, and additional comments about the feature. Although the feature collector form was not used for this system, it could be used to collect locations of important landmarks.

The data sent back to the server was sent to an ODK Aggregate client that I hosted on my Google Developers page. This enables multiple users to send data back to the server simultaneously. Furthermore, I could sign in to the client and export the data in various formats. For this project, I exported the records as a csv. Upon importing the CSV into QGIS I realized that the format of the trail path wasn't compliant with the WKT format that QGIS wanted the linestrings to be formatted as. The format of the nodes was exported from the GeoODK app as, 'lat1 lon1 elev1 acc1; lat2 lon2…'. In order to be WKT compliant the format would need to be, 'LINESTRING((lon1 lat1, lon2 lat2,…))'. My solution was to write a Python script that would iterate through all tuples in the CSV and convert the 'trail_path' field to the WKT format and write the new tuple to a new CSV. The final output was a CSV that could be imported into QGIS and then exported as a shapefile that I could then add into my geodatabase.

Since I knew I was going to ultimately use the OpenStreetMap background as my base layer I decided to download the path data from their servers to compare that data with the data I had already collected. The downloaded data was taken from the Java OpenStreetMap Editor and exported as a GPX. The GPX was then imported into QGIS and exported as a shapefile to be added to my geodatabase. Looking at the data, it appears that some of the data was taken from the University's trail data and some was just digitized in.

Now, I had all of my data in my geodatabase, including: 1) 'UMaineData_BikeTrails', 2) 'StravaRidesData', 3) 'GarminRides', 4) 'ODKData_TrailInfoCollector', and 5) 'OSMData_OronoTrails'. The next step was to

go through all of my datasets and pick the best paths for each trail by comparing it to the other sources and aerial imagery. I could usually see parts of the trails in the aerial imagery so I could then use this to assess which path is the most accurate. Generally, the 'ODKData_TrailInfoCollector' and 'GarminRides' datasets were the most accurate for the singletrack and main trails respectively.

I then made a feature dataset for my 'FinalData', which included the feature classes: 1) 'FeatureData', 2) 'IntersectionData', and 3) 'TrailInfo'. As I went through the data I would cut and paste the 'best' representative path for each trail in my 'TrailInfo' class, filling in the appropriate attributes for each feature. At the end, I went through and double checked for topological errors and cleaned up any mistakes I made while digitizing. The result was a nice clean trail system layer that had all of the trails in the portion of the DeMeritt Forest behind the University, including all of the attributes needed for my singletrack bike trails.

At this point, I started adding features to my 'IntersectionData' layer, using the UMaine DeMeritt Forest PDF as a baselayer. I added all of the features on the PDF map to my dataset and entered all of the data I needed for each intersection. This included adding a hyperlink column for the pictures of each intersection. I was unable to get pictures for each intersection, but I downloaded some sample pictures that would be used to show the functionality of the system.

With all of my data complete, I decided to return back to the Java OpenStreeMap Editor program so I could make the changes needed to get the OSM Baselayer to match up with my trails. I first had to adjust my 'TrailInfo' layer to include some of the attributes that are required by OpenStreetMap in order to style paths correctly. This required me using the OpenStreetMap map features reference to determine how to tag each path accordingly (http://wiki.openstreetmap.org/wiki/Map_Features). I then exported the 'TrailInfo' layer as a shapefile that could be imported in Java OpenStreetMap Editor. From here, I deleted all of the previous trails from the OSM servers and pasted my trails in their places. After replacing all of the old trails with the new trails I then had to go through all of the trails to make sure that all of the tag fields were properly populated from the shapefile. Once everything was finalized, I uploaded the changes to the OSM server. This resulted in an OSM baselayer that lined up perfectly with my TrailInfo layer.

I then could style all of my trails appropriately in my ArcMap project to create my final output map and I had baselayer that matched up with my data. I exported the PDF from ArcMap with the maintained georeference information so users could use the PDF for navigation on their phones. As discussed earlier, this wouldn't allow users to acquire information about the trails and it could not be extendable to other trail systems, so I had to start thinking about how I was going to get to the end result I wanted.

I wanted to take my local trail information system and convert it to a relatively simple interactive web app that easily be used on a mobile device. After some preliminary research early in the semester I determined that using Leaflet would be the best choice for my needs. Leaflet is an open source JavaScript library designed for creating interactive web maps and is implemented through CSS and HTML. Compared to some of its alternatives it is considered lightweight, simple and flexible. It is very similar to OpenLayers, but is considered to be a more lightweight and easier to use alternative. I even made a test page using the Leaflet library at the beginning of my project, which I included in my initial progress report, to see how practical using the library would be. The idea was to build the basic framework with the basic functionality I needed with the OpenStreetMap baselayer. Ultimately, I was happy with the result, and decided to continue using Leaflet. Now that I had my data I could add my layers to the test map I made earlier. I exported my 'TrailInfo' and 'IntersectionData' layers as shapefiles so I could import them into QGIS. From QGIS, I exported the shapefiles as GeoJSON files that could then be easily used in the Leaflet map. I then went into my HTML code and starting creating the styles that would be used on my web map. Once I got my styles and map layout finalized I went back to add some more features. I decided I wanted to add a measure tool and a PDF export function. The measure tool could be used if a biker wanted to know the

distance between points. The PDF export function would export my georeferenced PDF map that I created as my final output. This would be useful if I rider did not have a reliable internet connection, but they still wanted the navigation functionality of the system. I also tweaked the interaction with the data by adjusting the popup table to only include pertinent information. This would make the map less cluttered on mobile devices. I also adjusted the legend to be hidden by default. Since this was my first attempt at building a web GIS, each step discussed above required me to dive into the documentation for the Leaflet library to figure out how to get the output I wanted (http://leafletjs.com/reference.html).

Results

The final results of my project was an updated map of the DeMeritt Forest trails, including all of the new bike trails with their respective IMBA trail ratings, and a full-featured web map that could be used for navigating the trail system and getting information for each trail. The web map was hosted on my personal domain so that it could be accessible by anyone with an internet connection.

Discussion

Overall, I think the project was a success and I am very happy with the result. Improvements to this system would include updating the intersection pictures to be of the actual intersections. I'd also like to use the 'Feature_collector.xml' form I created to collect important landmarks and present them in the web map. Furthermore, I think adding the ability to add updates on conditions of trail systems would be really useful for local riders. Adding these additional features, as well as extending the system to include other trail networks, would require serious reconsideration of the design of the system as well as the layout of the web map.

My initial thoughts are to use the features from OpenStreetMap as the features on my map and extract the data for each feature from the appropriate OSM tags. The extracted data would then be converted to use in my map. This would prompt bike users to become more active in adding updates to the OSM project. I could then have a separate database for features, intersections, and trail updates that would also be included in my web map, but separate from the OSM efforts. The PDF output would then be automatically generated by extent of the users view.

Appendix

Data sources (for SIE509_BikeTrails.mdb)

1) UMaineData_BikeTrails09 (Feature Class) - *Supplied from the UMaine Forestry Department.*
2) StravaRidesData (Feature Dataset) - *Collected using Strava app on mobile phone.*
    a. StravaRides_UMOTrailSearch_10_05_15 (Feature Class)
    b. StravaRides_UMOTrailSearch_10_06_15 (Feature Class)
    c. StravaRides_UMOTrailSearch_10_07_15 (Feature Class)
    d. StravaRides_UMOTrailSearch_10_08_15 (Feature Class)
    e. StravaRides_UMOTrailSearch_10_21_15 (Feature Class)
    f. StravaRides_UMOTrailSearch_10_26_15 (Feature Class)
    g. StravaRides_UMOTrailSearch_11_04_15 (Feature Class)
    h. StravaRides_UMOTrailSearch_11_06_15 (Feature Class)
    i. StravaRides_UMOTrailSearch_11_11_15 (Feature Class)
3) GarminRides (Feature Dataset) - *Collected using Garmin 60cx GPS handheld.*
    a. GarminRides_10_26_15 (Feature Class)
    b. GarminRides_11_04_15 (Feature Class)
    c. GarminRides_11_06_15 (Feature Class)
    d. GarminRides_11_11_15 (Feature Class)
    e. GarminRides_11_12_15 (Feature Class)
4) ODKDataTrailInfoCollector_11_23_15 (Feature Class) - *Collected with GeoODK Collect app on mobile phone.*
5) OSMData_OronoTrails (Feature Class) – *Downloaded from OpenStreetMap servers.*
6) FinalData (Feature Dataset) - *Aggregate of above data.*
    a. FeatureDate (Feature Class)
    b. IntersectionData (Feature Class)
    c. TrailInfo (Feature Class)

Data dictionary (FinalData)

Trail Info <- MultiLine layer for my consolidated datasets

Field Name	Field Type	Field Width	Field Description
OBJECTID	ObjectID	----	ObjectID that is automatically generated for each record as it is inserted into database.
name	Text	254	Trail names as designated by the UMaine signage
management_grp	Text	25	Management group information
trail_system	Text	25	Trail network name. Only one for this project will be 'Umaine DeMeritt Forest'.
tr_system_town	Text	20	Trail system main town
tr_system_state	Text	20	Trail system main state
highway	Text	50	Used for OSM Classification
trail_type	Text	20	Type of trail that the data is being collected on. Includes: single track (ie mountain biking), ski trail, major trail, dirt road, or paved road.
trail_length	Double	10	Length of geometry in meters.
foot	Text	10	Used for OSM Classification
bicycle	Text	10	Used for OSM Classification
trail_widt	Short Int	2	Trail Width: criteria #1 for IMBA Trail rating system. 1=White Circle, 2=Green Circle, 3=Blue Square, 4=Black Diamond, 5=Dbl. Black Diamond
tread_surf	Short Int	2	Tread Surface: criteria #2 for IMBA Trail rating system. 1=White Circle, 2=Green Circle, 3=Blue Square, 4=Black Diamond, 5=Dbl. Black Diamond
avg_grade	Short Int	2	Average Grade: criteria #3 for IMBA Trail rating system. 1=White Circle, 2=Green Circle, 3=Blue Square, 4=Black Diamond, 5=Dbl. Black Diamond
max_grade	Short Int	2	Max Grade: criteria #4 for IMBA Trail rating system. 1=White Circle, 2=Green Circle, 3=Blue Square, 4=Black Diamond, 5=Dbl. Black Diamond
ttf	Short Int	2	Technical Trail Features: criteria #5 for IMBA Trail rating system. 1=White Circle, 2=Green Circle, 3=Blue Square, 4=Black Diamond, 5=Dbl. Black Diamond
overall_rating	Short Int	2	Calculated rating based on the above five categories. 1=White Circle, 2=Green Circle, 3=Blue Square, 4=Black Diamond, 5=Dbl. Black Diamond
IMBARating	Text	10	Rounded overall_rating. Text needed for styling purposes
details	Text	50	Various trail details can be added here.
pets	Text	20	What pets are allowed

Intersection Data <- Point layer for the major intersections marked in the Forest

Field Name	Field Type	Field Width	Field Description
OBJECTID	ObjectID	----	ObjectID that is automatically generated for each record as it is inserted into database.
management_grp	Text	25	Management group information
trail_system	Text	25	Trail network name. Only one for this project will be 'Umaine DeMeritt Forest'.
town	Text	20	Trail system main town
state	Text	20	Trail system main state
intersectionID	Text	50	IntersectionID assigned by management group or assigned manually.
intersect_trails	Text	100	List of intersecting trails.
image_hyperlink	Text	100	Image hyperlink for intersections

Feature Data <- Point layer for the major features to be marked in the Forest

Field Name	Field Type	Field Width	Field Description
OBJECTID	ObjectID	----	ObjectID that is automatically generated for each record as it is inserted into database.
management_grp	Text	25	Management group information
trail_system	Text	25	Trail network name. Only one for this project will be 'Umaine DeMeritt Forest'.
tr_system_town	Text	20	Trail system main town
tr_system_state	Text	20	Trail system main state
feature_name	Text	25	Name assigned to feature.
feature_desc	Text	50	Detailed description of feature.
image_hyperlink	Text	50	Image hyperlink for feature.