Web Map Assessment Based on Extracted GPS Points

February 1, 2017

Caleb Henderson
Shreejan Paudel
Brad Zecchino

Accuracy of GPS Readings Recorded with Each Method

The ArcGIS Collector App on both iOS (iPhone 5C) and Android (Nexus 5) devices, supplemented by Notepad recordings (place, time, and type) were the primary sources of data collection for Part 1 of the assignment. After scrutinizing data results, the accuracy of all devices seem to be quite minimal even at the furthest deviation from the hazard. The relative differences between collection accuracy on an Android and iOS device are minute, and essentially interchangeable. Accuracy variations could also be rooted in differences between device models and/or service providers, both of which could also be contributing factors to any identifiable deviation from the expected point locations.

It bears importance to state that this deviation may very well be the result of purposefully avoiding the hazard being recorded. For example, the danger of standing on a sheet of ice during data collection was outside our scope of commitment to Part 1 of the assignment. This source of error along with others, will be further discussed later in this report.

Ease of Usability

The ArcGIS Collector App is a relatively straight-forward, easy to use method of point data collection for people with GIS and mapping expertise. Once a map has been set-up for collection usage, the actual data collection is a simple, smooth process. The Collector App is quick and fluid, offering an easily navigable interface with speedy loading times. Uploading data proved to be very simple for offline clients when re-entering a WiFi zone, or even faster with automatic uploads for those working on a data network. These capabilities would serve to be more than adequate for any type of municipal staff attempting to record point data around a city.

Please note that these advantages are critically based on access to the ArcGIS online database and are not free. Any organization wishing to facilitate data collection through the Collector app and organize it through ArcGIS Online must obtain a license (at a fee) beforehand. This barrier will most definitely inhibit any ‘casual users’ from actively participating that might be available regarding road/community maintenance.

Hazard Domain Assessment

The hazard domain provided for the data collection of this assignment was passable for its intended usage, but by no means complete. The categories were adherent to the accessibility guideline resources that were referenced, including City of Ottawa, Kitchener, and Toronto. Regardless, many areas were identified as incomplete and raised significant room for improvement. Firstly, some hazards can be defined under multiple categories, but Arc Collector does not incorporate a multiple classification option. For example, the absence of a curb cut can also be classified as a tripping hazard, and excessive cross-slope of a sidewalk can also be categorized as excessive slope. A quick fix for this issue would be to narrow the classification options, and offer further explicit details on what qualifies for each hazard type, to completely eliminate these overlapping categories. Additionally, some categories that are situation-specific could not identified, for example a lack of lighting during naturally dark hours of the night could not be classified according to the provided hazard domain. A situation-specific category for the winter would also serve to be beneficial. This absence forced misidentification of conditions where an ice hazard could be identified, groups were forced to categorize these as ‘water-pooling’ where there is obviously no standing water. Additionally, physically long hazards such as icy walkways or consecutive missing sidewalk slabs could only be identified as single points, even if the hazards stretched for many metres, before multiple hazards points needed to be identified through the app. Line and polygon options for data collection would serve to rectify this complication, to avoid multiple points of recording for a single extensive hazard. Lastly, if the recorded data was collected with the intention of repairing the existing hazards, the domain provided categories are not in easily digestible terms for the repair parties. This data would be much easier to read, understand, and act upon for a service party if the hazards were classified based on the repairs necessary to void the hazard itself. For example, ‘Missing Curb Cut’ could be classified as ‘Concrete Removal Necessary’, and ‘Unevenness’, ‘Excessive Cross Slope’, or ‘Missing Sidewalk Slab’ could all be classified as ‘Asphalt Addition Required’. This simple change could save massive amounts of work for any employee assigned to deal with these hazards, as a result of this data collection.

Pros and Cons of Displaying Using Point Features

The Pros of using point features are based on the accuracy of the data collected. Each point is specifically finite in its location allowing incredibly specific identification of the recognized hazard in the field. This accuracy unfortunately also lends itself to a pretty considerable Con. If the data is subject to any type of error in the recording, the point itself is going to very specific to an incorrect location. Point data, however, lends itself very favourably to field recordings. The ease in which you can create a point feature over the multiple nodes of a line or polygon make this unit of measurement ideal when in the field. Another benefit is the capacity for comparison between other points. There is no need to compare feature geometry when every created feature in the data set has the same dimensions, a 1x1 point. That being said, long features can only be identified as a point, or multiple points, both of which do not effectively represent the issue of an extended feature. It is also important to note that not all points are equal, because not all hazards share the same magnitude. When points are being compared, it is difficult to identify the severity or presence of the real-life feature it represents. For example, in the hazard point collection for this assignment, both a fallen-tree and cracked sidewalk slab are identified with the same feature, a point, but offer considerably different hazards to people walking each path.

Challenges Encountered

Challenges encountered when integrating hazard points collected by many different users are mostly dependent on human inconsistencies and errors. With every group of data collectors, each individual will has different standards for each ‘hazard’, even if the categories have been explicitly defined by the governing body, the identification of the hazard is still up to the discretion of the individual. Moreover, each user has a unique understanding of the data collection tool, and may not know about a certain functionality available. This combined with the differing levels of scrutiny that each point will be collected with, will inevitably lead to some serious inconsistencies within the shared data set. Lastly, if multiple users are covering a shared area without finite and easily understood boundaries, as in the field, multiple recordings of the same point are definitely possible, and will need to be screened out after data collection.

Evaluating Four Different Web Mapping Applications

Development Environment

ArcGIS Online offers a basic step-by-step approach to building a web map. Organizing and displaying data is made simple through the default display settings, however there is quite a limited amount of advanced functionality. Any type of analysis or data management further than the standard symbology manipulation and basemap customization is out of reach. However, these limitations are not binding considering how tightly WebAppBuilder is linked to ArcGIS online maps. WebAppBuilder is very approachable a “plug and play” environment for casual users. ESRI JS-API is more complex, yet still straightforward in developing a basic map. It also has more than one way of creating maps. The complexity seems to rise quite a bit for advanced maps, some functions seemed to work better together than others when implementing. Leaflet offers easy to use functions which can be used to create a fairly powerful web map. The documentation was fairly well done for Leaflet but the tutorials on how to integrate different functions together are limited. All three products have good documentation in its website and has good contribution of examples from 3rd party websites/forums as well.

Leaflet and ESRI Javascript API both had coding involved in making the maps, however, they were slightly different when viewing. When using basemap, the street-map of Leaflet was different from ESRI, therefore, we implemented the satellite view on both maps. Leaflet has a single legend library that can be called, whereas for ESRI, it has to be implemented through various library calls. ESRI seems to have more customization with more complexity, whereas leaflet has easier way to implement but less customizations.

Organizational Requirements

There is a free 60-day trial for ArcGIS Online, however any extended period of use demands a user pay for full access to the complete catalogue of capabilities offered. WebAppBuilder requires a link to an ArcGIS Organization. While it’s difficult to tell the exact price of this link it is a safe assumption to say that it would be expensive. Licensing for ArcGIS Javascript API states that there is no fee for using the API or publishing any application built from it. However copyright from ESRI states that the Web API’s can only be used in conjunction with ArcGIS Online ArcGIS for Developers, Portal for ArcGIS or ArcGIS for server. The financial costs seem to be different based on the ArcGIS service being used. The Leaflet API has no cost. Any ERSI platform offers a considerable option for internal data use. An easily accessible and organized interface like ArcGIS online or WebAppBuilder would serve great use to subsections of municipal government offices who are out-of-touch with GIS and its extended capabilities. The City of Waterloo would have the ability to implement it despite the cost, as it offers many applications without having to code each one separately. A startup though, may not be able to justify the cost and would rather use an open source option like the Leaflet API.

Functionality

The display capabilities of the collected ‘Sidewalk Hazard’ data are sufficient and reliable through ArcGIS Online and WebAppBuilder. All the functionality necessary was available and easy to use. The graphic widget in WebAppBuilder is a very interesting tool but was found to be next to useless due to it’s inability to properly display issue names, meaning an end user would probably just find it rather ugly looking. ESRI Javascript API made the tasks relatively easy to accomplish however, the legend of the map caused difficulties because it created a separate html box outside of the map. Leaflet API had the capabilities needed to complete the task, but not to the highest degree they could be. Filtering/toggling layers on and off was not an option primarily due to the way in which the geoJSON data was stored. Leaflet is great for basic functionality to make a great looking map to just display data points, but any upper level functions are easier to implement elsewhere.

Performance

The small-scale neighborhood size of the assigned area makes ArcGIS Online a breeze to load and pan in a browser. Larger scale map manipulation and greater dataset size could jeopardize the speed and fluidity of the ArcGIS Online Map in question. The most disappointing aspect of WebAppBuilder was the performance. Every time you load in the map, it first adds in the map, then the points and finally the windows. The points take quite a while to finish displaying. Each time the map is reloaded it has to regenerate everything again, so it's doubtful anything is stored in your cache to decrease loading times. ESRI Javascript API takes time at the beginning to open, however once loaded, it is quick to adjust when zooming in/out. Leaflet posed issues when loading multiple base maps, as it tends to load tiles one at a time initially on startup. Considering the small amount of data used in this assignment, it is very unlikely to see any large performance issues. Nonetheless, larger maps with more data points will take significant amount of loading time.