Developing a Low Cost, Offline GIS for Community Mapping in Myanmar

Alecio O'Day and Bobby Sayers
GEOG 569 GIS Workshop
17 August 2018

A Technology and Social Change Group (TASCHA) Project
University of Washington

Executive Summary

Background

This project was undertaken to support the United Nations Sustainable Development Goal #5, "Achieve gender equality and empower all women and girls". The Technology and Social Change Group (TASCHA) at the University of Washington proposed the development of a low cost, offline Geographic Information System to help empower communities in Myanmar, successfully funded by the Bill and Melinda Gates Foundation in the amount of $100,000. Their proposal has community members recording instances of gender inequality by taking photos with a cellphone, adding descriptions to them, then sharing and visualizing the information on a Raspberry Pi computer in a local library setting. With no anticipated internet capabilities in Myanmar, this needs careful attention as to planning and preparation to ensure successful deployment.

Through the Master of GIS for Sustainability Management program at UW, we worked under guidance from TASCHA to help develop this Raspberry Pi-centered configuration. Specifically, TASCHA asked us to look into whether a Raspberry Pi could host the open source software QGIS and act as the central repository and local hub, to be scaled to meet different geographic areas and topical needs. Additionally, could we identify and possibly create a workflow for capturing geotagged photos and importing them to this GIS for visualization by the community? And in regard to data management and scalability, could we develop a database to host this public participation GIS data? If any or all of these requests were possible, would we then be able to document the process for future replication and test our work by physically configuring one of their Raspberry Pi units?

Issues

Several small but vital issues were littered along the development path but there were some large, looming questions from the onset. First, everything would have to be considered with no potential for internet as a fallback. All the ease of connectivity and the quickness with which information could be harnessed (with web-GIS software like Google Maps) had to be shed from our possible solutions. A major issue was whether QGIS would even run on a single-board computer. Integral to the workflow was how to embed text-based descriptions into the metadata of the images so that this essential information could be passed along in the importation process. Collecting spatial data means nothing if it can't be geo-referenced. How would we make the data meaningful in a spatial sense without an internet-derived basemap?

In terms of work scheduling and progress, we had to ensure that we worked in parallel, assimilating each other's development work to best move forward. In reality we were employing a versioning structure workflow to ensure replication. As we see in the outcomes of the project, we were able to stay in sync, working together to provide solutions for these issues.

Outcomes

Overall, we were able to provide answers to the questions posed. We were able to load and run QGIS with little trouble and compose a SpatiaLite database to host the collected data. We identified three valuable QGIS plugins to import (*ImportPhotos*), visualize the images (*eVis*), and supply offline basemaps (*QOSM*). The first of these needed our Python coding to support the embedding of description field to the metadata. This allowed for the image to be visualized in QGIS along with its description and other vital data such as time, date, and name. We also tested the upload speeds for increasing batches of photos, looking for processing interruptions and crash issues. Our results include tables with estimates for photo file sizes; basemap file sizes with regard to extent and scale; and the database schema built to accommodate the photo metadata.

In terms of deliverables, we were able to detail our process in the form of an installation guide. This document establishes a workflow and methodology for getting the Raspberry Pi setup from an office in Seattle to an offline community in Myanmar. We then used this documentation to successfully configure the TASCHA supplied unit, which we will return to them at the on-campus meeting. Although the process was not without setbacks, we feel that our work will at least provide a framework from which to build upon, if not be implemented somewhat close to as-is. We make recommendations in the report's conclusion to better aid TASCHA and the communities of Myanmar in fully actualizing this technology as it supports the cause of gender equality.

Table of Contents

9. Technical Appendices (as appropriate)
    9.1. Appendix A - Additional Resources
    9.2. Appendix B - Raspberry Pi specifications
    9.3. Appendix C - QGIS requirements

1. **Background and Problem Statement**

This section details the project background and problem statement, enumerating the "need to know" questions in the context of the *MGIS-GEOG 569 GIS Workshop* course.

Our project is to develop and test a low cost, offline GIS capable of hosting geotagged photos to support communities in Myanmar and Sub-Saharan Africa for the Technology and Social Change (TASCHA) Group at the University of Washington. It was proposed with the intention of using a *Raspberry Pi* single-board computer and the open source GIS software *QGIS* to allow community users to document instances of gender inequality. This would be done by taking a photograph with a cellphone camera, adding a text description, and having the GIS ingest the user data for interaction in a pre-identified location such as a library or community center. TASCHA was able to secure a grant to support this work through the Bill and Melinda Gates Foundation. Jason Young and Chris Jowaisas are the contacts at TASCHA for this project, proposing the enlistment of MGIS assistance and overseeing our development work.

In support of the United Nations Sustainable Development Goal #5, "Achieve gender equality and empower all women and girls", TASCHA (Gates Proposal 2018) aims to support gender equality by allowing community members to share photographic information about personal experiences regarding gender inequality. Because of the lack of widespread internet availability in places such as Myanmar and Sub-Saharan Africa, a normal web-based GIS solution (ArcGIS Online, Carto, OpenStreetMap, Google Maps) is not suitable. Therefore, an offline solution is needed to enable community members to map these instances and share them within the community. In their proposal to the Gates Foundation, TASCHA states that creating a platform such as this can allow citizens to "contextualize global problems within their own lives, visualize pragmatic solutions to those problems, and then collectively implement solutions."

1.2 Need-to-Know Questions

Items identified as need-to-know questions included whether the Raspberry Pi single-board computer is a suitable option for serving as the hardware in a GIS. With a small footprint and a very affordable price-tag, the Raspberry Pi computer is widely used for hobbies, especially supporting inventive electronic and IT-related endeavors. Whether it could handle the needs of a Geographic Information System and act as the central repository for potentially large photo batches was in question. If indeed possible, the follow up question would be, could it be scaled easily to meet multiple disconnected and discontiguous environments? In other words, could nearly identical Raspberry Pi setups be deployed in separate offline areas such as Myanmar and Sub-Saharan Africa, with the only large difference being the community it serves? Lastly, how easily could a GIS database be built on this hardware to host geo-tagged photos for visualization by the community members? What would be necessary to complete the process of geotagging photos and visualizing them offline? How would users add personalized descriptions of the images and how would the images be configured in the database? If successful, could we provide a step-by-step guideline of the process? Further, could we outfit a Raspberry Pi for TASCHA with our prototype configuration? These need-to-know questions became the basis of our development work and detailing our successes and failures would be just as informative. Documenting this process would assist in future development of the larger proposal.

2. **System Resource Requirements**

This section describes the system resource requirements including all data needs, software, hardware, network capabilities, personnel, and institutional resources necessary to facilitate the project.

Data resource needs for the scope of this project require a database containing three entities: a cellphone photo of a point of interest, a related point layer representing its location, and a geo-referenced basemap image for geographic orientation of the point data. The photo is to be an instance of gender inequality in time and space, interpreted by the community member who takes the photo. By enabling the location services of the cellphone, the image will automatically be stamped with valuable metadata such as a name, date, time and geographic coordinates. A crucial component of our responsibility is adding a text description field that can be appended to the metadata so that this user-generated information can be seamlessly imported into the GIS. The coordinate information from the metadata will be used to create a point layer that can be visualized in a GIS by community members. This point data will be stored in a SpatiaLite geodatabase to align the metadata schemas and allow for a central storage repository on the Raspberry Pi unit. If one were to use the shapefile format, schemas from multiple files could become unaligned. Providing geographic reference for the vector data is the basemap imagery that will need to be pre-loaded before being set up in an offline environment.

2.1 Software capabilities are minimal yet standard for a GIS. Simple processing is needed such as the importation of photos that creates vector data in the form of a point layer. Visualization of the features and their attribute information is necessary (the photo description must be accessible) as is a geodatabase to host the data and maintain its integrity over it lifetime, all standard components of any GIS. With these standard necessities, TASCHA has suggested the free and open source QGIS program as the basis for software testing. Being the most widely used and supported open-source GIS, we can be fairly confident that this is the recommended choice, especially since expansion of the offline setup could span large geographic regions and be deployed for up to a year without ever using a web connection. The most robust open source software that keeps overall costs down is by far QGIS. Although other softwares may be able to handle the simple requirements of the project, no other open source GIS has the developer support and widespread implementation of QGIS. Whether there are any roadblocks in using this software is another question we are charged with answering.

2.2 Hardware requirements are dictated by the developmental nature of the project, as TASCHA is asking whether the Raspberry Pi single-board computer can host QGIS, the processing and the storage of the data. Instead of a project where we investigate any number of hardware solutions to fit the problem, this project is unique in that a hardware solution has already been identified to be tested. Our responsibility will be documenting whether this is hardware is suitable for the project. The Raspberry Pi setup we are tasked with investigating is *Model B+*. It is equipped with a 16GB mini SD card that hosts the Raspbian Operating System. The 64-bit processor has a speed of 1.4GHz with 1GB of RAM. It has built in ports for HDMI (display), Ethernet, USB (4 ports), and is Bluetooth and WiFi capable (2.4GHz and 5GHz 802.11b/g/n/ac). Although a web connection is necessary in testing and in the initial setup, once in the field the unit is to be used offline.

2.3 Data input storage requirements will be determined by 1) the number of photos collected and transferred, and 2) the storage size of the basemap tiles. There is no limit to the amount of instances of gender inequality a community could record. Since they are recorded in the form of photos with metadata, this could become infinitely large. The project is to last roughly one year, with TASCHA estimating somewhere between 20 and 100 photos being transferred per person per week. It is unknown whether this rate will stand over the course of a year or not. However, if using the low estimate of 20 photos/week, one year would accumulate roughly 2.4GB of photos. At 100 photos/week, this estimate rises to about 11.6GB. Although both fall within the 16GB threshold, basemap tile storage still needs to be accounted for. These estimates also do not consider whether any photos will be backed up/archived using other storage means, or whether any quality assurance would pare down these weekly estimates. Basemap tilling can vary widely also, and is highly dependent on the extent of geographic coverage and the scale of the imagery. In our testing, a highly localized basemap at a scale of 1:2500, covering roughly 0.5 square mile, used 67 KB of storage. This was a QGIS OpenStreetMap basemap of the variety commonly used in the default version of Google Maps, with street names and other prominent features labeled. A coarser scaled basemap at 1:10,000 covered about 2 square miles and used 137 KB of storage. The level of basemap granularity needed to effectively visualize the data should be considered carefully. Since the overall project is focused on Myanmar at national level, it is difficult to speculate how much of the country will need fine scale basemap imagery. It is uncertain how many Raspberry Pi setups might be deployed in the country or in any one city. If there are multiple, the storage of fine scaled basemaps could be shared across the units. On the other hand, if only one setup is used for the entire country, the storage capacity of one 16GB memory card will be quickly filled up if attempting to create tiles of varying scales and extents. For a table detailing storage estimates for photos and basemap, see Appendix A.

2.4 Data processing storage requirements are minimal. The processing needs include the data importation, which creates a GeoJSON file, and the editing/appending of data to the SpatiaLite database layer. In our testing, the 1.4GHz and 1GB RAM processor easily imported batches of one thousand photos in under 1 minute. Batches of twenty photos took under 2 seconds to process. Editing and appending data to the SpatiaLite layers was not timed as this takes much less processing power when done in batches of 20 and 100 at a time, as this is the normal workflow. The test SpatiaLite database hosting the vector data of twenty -two photos consisted of 32.1 KB.

To import the location and description data into QGIS, we modified the Python code of an existing plugin (*ImportPhotos*) under the terms of the GNU General Public License. The unmodified plugin created a GeoJSON file with the name, date, time, latitude, longitude, and other location data. The primary modification that we made to the plugin added the ability to read the ImageDescription EXIF tag (which could be populated on the original picture by using metadata editing apps on a mobile device), and write it to a Description field in the GeoJSON file. Subsequent updates to our plugin improved its error handling capabilities; when it encounters a picture without location or description information in the chosen input folder, it will simply avoid importing it and move on to the next picture instead of ceasing the script at that point. Error messages were also changed to reflect the absence of both location and description metadata.

2.5 Data output storage requirements solely consists of a GeoJSON as the output file for every importation. An estimate of a GeoJSON containing twenty points used 9.0 KB of storage. These files are relatively small (since they contain the file path to the imported photos, not the photos themselves) and we don’t anticipate them having a large influence on the overall storage needs.

2.6 Personnel resource requirements were acquired through the project sponsor in the form of a budget breakdown (TASCHA Misk Budget Justification 2018). No estimates of hours were provided in the document although a timeline of one year was identified as the lifespan of the project. In addition to our development work over the course of the workshop [which we estimated at somewhere between 180 and 270 hours (10-15 hours/week/person x 9 weeks)], TASCHA is budgeting four (4) personnel to carry out the implementation of the project. First, a supervisory role who conceived of the original proposal, will provide oversight throughout the project’s lifespan. Next is someone responsible for workshops and local(Myanmar) personnel training. A third person will be tasked with developing a training curriculum. The fourth member is the local organization Myanmar Book Aid and Preservation Foundation (MBAPF). MBAPF has a local knowledge of the Myanmar library environment and will be involved with translation of training materials. MBAPF could be considered an institutional resource requirement, but no others have been identified in this category (TASCHA Misk Budget Justification 2018).

2.6.1 Our own development work was planned out using a Gantt chart early in the quarter (see Figure 1). Many tasks were not reflected in the chart; some of our work came as the result of informal discussions after class or as individual experimentation, and was not planned in advance.

Figure 1. Gantt Chart

In the section that follows we will attempt to provide a cost-benefit analysis that details known costs and perceived benefits.

3. **Business Case Evaluation**

Here we build a business case, enumerating costs and benefits and looking at these numbers in the context of the lifespan of the project. First, a brief overview with context; then we list benefits, much of which are not immediately quantified in monetary terms; next we lay out costs for the project; and end with a cost-benefit analysis. Essentially, we lay out a case for implementing a GIS in place of a traditional method.

Some of the more tangible aspects for this project include hardware and software costs, as well as personnel. These components are itemized in proposal (TASCHA Misk Budget Proposal 2018). The Raspberry Pi computer used in our testing has a stand-alone price of around $35. Our efforts that fall under development can be quantified at least in terms of hours spent. Speculation will play a large role due to the nature of the intangible benefits, primarily promoting gender equality. Social change at this level can have lasting effects that can take many forms and play out over generations. The basis of this project is to increase social awareness and start dialog on the difficult, but necessary topic of promoting gender equality. Awareness and subsequent discussions can lead to changes in behavior that can ultimately affect particular businesses and influence choices about education paths, job options and even where one chooses to live. It is obviously difficult to understand how a life led in a less oppressive society is hard to quantify in terms of monetary value, but we can speculate on what kinds of economic effects might occur from more empowered decision-making (Elwood 2002). In doing this, we will attempt to make known our speculations and assumptions for this particular project as we understand it.

3.1 Benefits

To examine benefits, we will use two methods for itemizing a needs assessment. First, in addressing quantifiable efficiencies in current practice as laid out by Antenucci et al. (1991), we eliminate redundancy by creating a single, standardized repository to house a community's points of interest. If no standardized container was created, data could potentially be multiple files with mismatching schemas. Establishing a common coordinate system eliminates misaligned data and limits discrepancies. These types of nuances create extra work to compile data for the GIS, especially when a project needs to be scaled up to the national or regional level. Adding a text description to the photo while the data is still on the phone would otherwise have to be done manually after importation. In this latter case, the user would have to record notes in a notebook or another mobile app for data entry later. This workflow introduces the possibility of data degradation because notes may be written down wrong, accidentally forgotten or mismatched with other features. Adding a description immediately keeps the data bound to the photo throughout the process, limiting potential for errors.

In terms of additional staff equivalency, the GIS helps here as well. If no GIS existed for this project, it would take much more time for users to plot their points of interest precisely. In attempting this in analog fashion, a large physical (paper) map would be necessary and regardless of scale, additional personnel (of some technical ability) would need to transpose useful spatial information to this physical map. With digital capacity, all basemaps would be pre-loaded on the Raspberry Pi setup, prepared by quite quickly by someone with basic GIS knowledge. The benefit is that when accumulating data on many scales, the digital platform provides seamless compatibility among basemaps and data, which cuts out any additional staff requirements that might have to stitch maps together later.

The potential for unpredictable events is very possible in the form of social uprising and rebellion. Though the intention of this project is to spawn conversation about gender inequality, the potential for more spontaneous-type events is very real. The Arab Spring was just this, possible through increased spread of information using social media. Facebook and Twitter hosted unfiltered opinions, creating communication where it was absent previously (Kojoori-Saatchi 2015, 59). When used to communicate gender equality based-information, it is not difficult to imagine a similar reaction. Responses might be boycotting a local business; forming a local advocacy group; or seeing an increased participation in local politics to help shape laws.

Intangible benefits that might be provoked by such a project include increased happiness and decreased oppression. TASCHA's grant proposal states that the implementation of this project aims allow communities to "*contextualize global problems within their own lives*, *visualize pragmatic solutions* to those problems, and then *collectively implement solutions.*" Routes to achieve this could be different for participants, but it starts with sharing information about gender inequality. Would identifying a local market or business that practices gender equality be beneficial for persons experiencing discrimination? Yes, it could promote positive self-identity (Young and Gilmore 2013) and strengthen the community through acceptance of values. It could drive awareness of community members who don't necessarily experience gender inequality or that promote it, either intentionally or unintentionally. Local businesses might see change in patronization depending on whether they support or oppress gender-related issues. The state of North Carolina was recently boycotted by locals (Reuters 2017), celebrities and major companies over the what was called the "Bathroom Bill" (NCSL 2017). While Myanmar and the U.S. are quite different culturally, this type of change is not far-fetched.

In terms of the quantifiable sale of information, it is possible that at some point newly attributed geographic information could be leveraged towards internet marketing. If information regarding gender inequality were to reach social media platforms (not its intended use), public participants (using social media) could make choices about potential businesses based on this information. If these decisions were made public (as in a social media post, internet marketers could use this info to strategically place online advertisements to these individuals. While this is highly speculative, this type of sale of information does occur in other parts of the world based on these very personal decisions.

Using the Huxhold (1991) format, there two relevant components. Cost avoidance is the first. Instances of gender inequality are very personal so by nature it is necessary that the public participate to fully address this issue. Therefore, collecting data digitally is not avoiding costs in terms of who is collecting the data but with synthesizing trends across datasets. The building of a GIS overwhelmingly reduces costs of having to collect information about places via other means. Coupling photographic data with coordinates and text descriptions would take much more time and resources if done with physical maps, separate GPS units, notebooks, and written descriptions.
Gathering information on gender inequality instances would also likely lead to increased revenue in terms of increased participation of community members who may have been persecuted and oppressed because of gender. The incorporation of oppressed members of society is much more valuable economically as they become integrated in day-to-day economic transactions.

3.2 Costs

Cost for this project is high priority, as free and low cost items allow for the setup to be even more widely distributed. The hardware and software setup that we are testing will most likely be purchased in bulk, by component, within the limit of the Gates Foundation grant award totaling $100,000. The following outlines those costs as well as implementation and maintenance costs (TASCHA Misk Budget Proposal 2018).

Capital costs such as a database will have no charges, as we are using a *SpatiaLite* database, created in the open source software *QGIS*. Both of these softwares are free and consistently supported by the international GIS community. Hardware is made up of several small components. The *Raspberry P*i single board computer sells for around $35. Additionally, it needs storage capacity and an Operating System. A 16GB MicroSD card with the *Raspbian* OS costs $15. *QGIS* software is free and open source. A wireless keyboard and mouse sell for about $25 and $15, respectively whereas a monitor for visualization runs between $55 and $100. TASCHA plans to provide cellphones in cases where users do not otherwise have one. We estimate that a 2-3 year old *Android* platform cellphone roughly costs around $100 each. The proprietary nature of *iPhone* would be too difficult to include in the project. The mobile app *Photo EXIF Editor*, in which the text description will be added, is available free in the *Google Play* store on the mobile device. The basemaps for the project will be *Open Street Maps*, available via the *QGIS* plugin *QOSM*. These are also free.

Development work to build the proposal can be considered part of the personnel costs. Although it is unknown, it is unlikely that TASCHA assumed MGIS assistance was available in the form of development work. We estimated our hours at 180-270 total hours (10-15 hrs/week/person x 9 weeks). Therefore, these hours would be considered additional cost estimates.

Electricity costs in Myanmar for hosting data range near US$0.03/kWh (Myanmar Times 2018) and whether this will be assumed by the maintenance budget is unknown. We estimated the usage being around 1 hour per transfer (one transfer meaning one person passing their personally collected data to the database). It is unknown how many users will participate and how often. If the data was eventually accumulated and hosted by TASCHA (in Seattle, WA) the rate for electricity as of July 2018 would be around US$11.1/kWh (U.S Bureau of Labor, July 2018).

In the next section is TASCHA budget proposal.

3.2.1 TASCHA Proposed Costs: Total: $100,000

TASCHA identified the following costs:

Personnel: $28,772

Personnel costs covers salary for the Principal Investigator who provides project oversight. Two support staff are also included; one person to lead workshops and train personnel in Myanmar; and another who will design the workshop curriculum.

Contractor: $46,625

The identified contractor is Myanmar Book Aid and Preservation Foundation (MBAPF), a local Myanmar partner who will assist with multiple components of the project. They have local knowledge of the Myanmar library environment and have previously provided a solid performance in prior engagements with TASCHA. They will provide translations for the curriculum and trainings, as well as helping the process development. They will also provide ongoing support to program participants and participate in evaluation activities.

Travel: $22,920

Travel costs cover travel for two(2) project personnel to Myanmar three(3) times to facilitate workshops, install the technology and train personnel. Covering 6 trips total, a single trip is estimated at $3820.

Supplies: $1,683

Supplies are considered the hardware and software, consisting of Raspberry Pi computers, monitors, keyboards, computer mice, and cellphones.

Table 1. Cost table reflecting TASCHA proposal budget

Cost Category	**Total Cost ($)**	**Total Cost in Percent**
Data Collection: community	$ -	0.00%
Management and Overhead (TASCHA)	$ 28,772.00	28.77%
Hardware and Software Maintenance (listed as Travel:2 personnel x 3 times to train/install)	$ 22,920.00	22.92%
Contractor (Myanmar Book Aid and Preservation Foundation (MBAPF)	$ 46,625.00	46.63%
Hardware and Software (Initial Purchase) listed as Supplies in their documentation	$ 1,683.00	1.68%
Support Staff (MGIS students' work)	$ -	0.00%
Other	$ -	0.00%
Total	$ 100,000.00	100.00%

3.3 Cost-Benefit Analysis

Due to the multinational nature of our project, we had to take into account the differences in labor cost in the United States and Myanmar. If we strictly performed a dollar-amount comparison between the cost of our labor for this project and the cost to achieve the same goals in Myanmar without our project (using local mapmakers rather than GIS software), it would not appear to be financially viable to utilize our workflow. For this reason, we calculated the cost for this quarter in terms of hours. Additionally, many of the benefits enumerated in section 2 take place on a much greater temporal scale than the 9 weeks of our participation. (This will be covered in section 3.4 below.) In Table 2, we've compared our weekly tasks with the predicted equivalent tasks without our project. We then estimated our weekly cost in hours, which accounts for our time as well as the time TASCHA personnel spent meeting with us and creating the initial project presentation, and compared it with the estimated weekly costs without our project. Table 3 also has columns for the cumulative weekly cost with and without our project. Figures 2 and 3 display these numbers graphically, and it becomes apparent that many more hours have to go into training community volunteers to do tasks that open source GIS software can accomplish effortlessly (particularly in creating basemaps and manually displaying the collected data).

3.3.1 Cost-Benefit Analysis Tables

Table 2. Comparison of Weekly Tasks

Week	**With Project**	**Without Project**
1	Receiving partner presentations	Receiving partner presentations-identify organizer
2	Reviewing literature, getting Raspberry Pi up and running with QGIS, Scope of Work, meeting with TASCHA	Determine Scope of Work, research options for recording coordinates
3	First attempt at workflow, researching alternatives, meeting with TASCHA, documentation, data design	Identify methods for acquiring GPS/camera assets within community
4	Identify Android app for editing photo metadata, began editing ImportPhoto plugin, saving basemaps for offline use, meeting with TASCHA	Train community volunteers in documenting gender inequality via pictures, coordinates, and descriptions
5	Progress with ImportPhotos, SpatiaLite database, proved viability of workflow, Business Case assignment	Find community member who can create paper basemap, continue training from previous week
6	Modified ImportPhotos plugin now stable, Business Case document, mostly done with development, meeting with TASCHA	Test workflow by taking and printing photos, posting/visualizing on paper map with organizer
7	Build out the prototype RPi for TASCHA	Assess results of workflow(organizer)
8	Create documentation for referencing build out and processes	Finalize documentation for workflow
9	Finalize documents and reports	Start field use of the workflow

Table 3. Cost Comparison (in hours)

Week	**Weekly Cost of Project**	**Cumulative Cost of Project**	**Weekly Cost w/o Project**	**Cumulative Cost w/o Project**
1	40	40	40	40
2	25	65	40	80
3	35	100	80	160
4	35	135	120	280
5	35	170	120	400
6	35	205	40	440
7	25	230	20	460
8	25	255	20	480
9	35	290	30	510

Figure 2. Weekly Cost (With and Without Project)

Figure 3. Cumulative Cost (With and Without Project)

When we extend the cost-benefit analysis across the full timeline of TASCHA's project (field work scheduled from February to August 2019, as well as the months of development, preparation, and analysis surrounding the field work), the benefit of our project becomes more evident. These estimates were made using a similar methodology to our short term cost-benefit analysis, for the phases of the project listed in Table 4 below. Figures 4 and 5 show the quarterly costs and cumulative costs with and without our project. Once training of the volunteers is complete, we predict with high confidence that performing the fieldwork and analyzing the results without the use of GIS software will take a much larger amount of resources than our workflow. Just like with the shorter term described above, the ability to quickly take photos, write descriptions, import the photos, and display them on a basemap will enable a greater amount of participants to collect data, and for that data to be aggregated more efficiently across a region.

Table 4. Cost Comparison (in hours), long-term

Quarter	Phase	Quarterly Cost of Project	Cumulative Cost of Project	Quarterly Cost w/o Project	Cumulative Cost w/o Project
Q2 2018	Development	65	65	80	80
Q3 2018	Development	225	290	430	510
Q4 2018	Preparation for Fieldwork	150	440	200	710
Q1 2019	Preparation, Fieldwork	200	640	400	1110
Q2 2019	Fieldwork	300	940	600	1710
Q3 2019	Program evaluation, analysis, and write-up	200	1140	400	2110

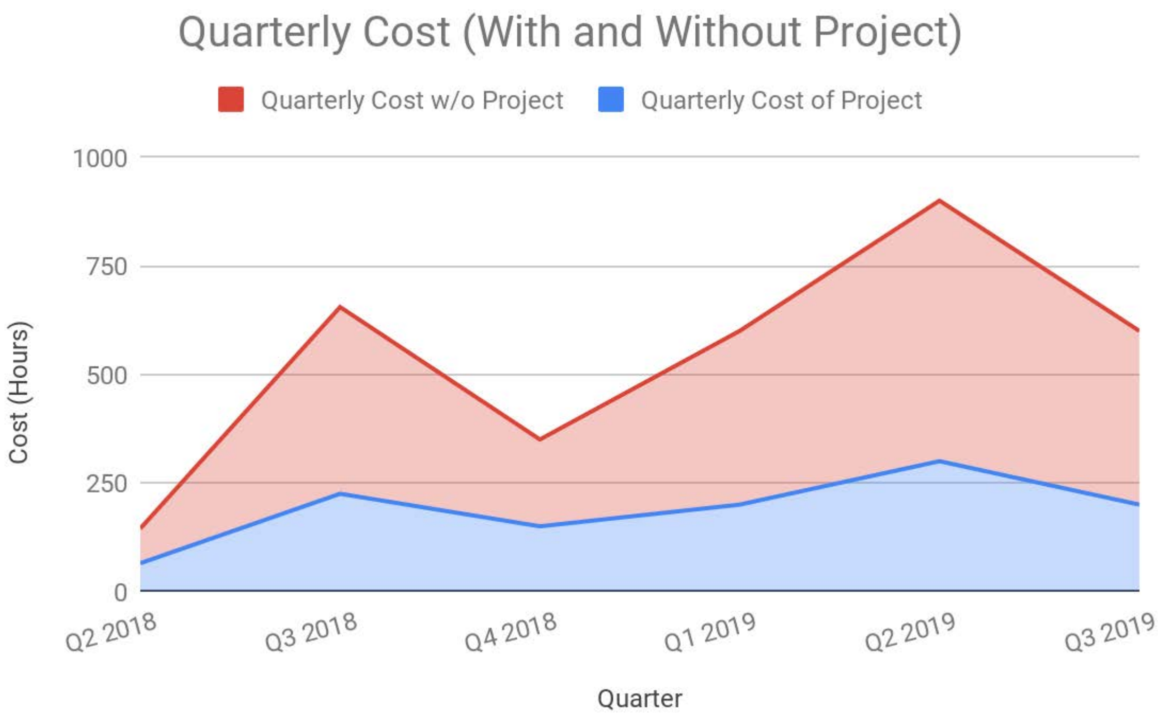

Figure 4. Quarterly Cost (With and Without Project)

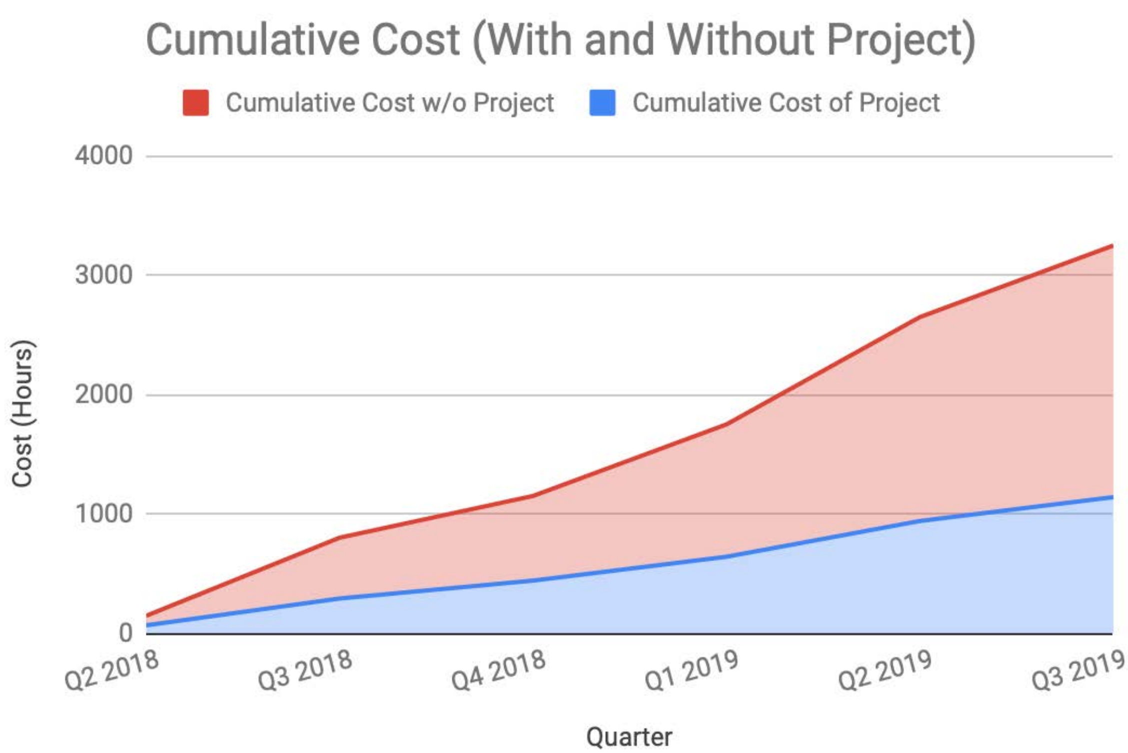

Figure 5. Cumulative Long-term Cost (With and Without Project)

In the next section we look at the Data Design process.

4. **Data Development**

In this section we will provide insight about the process through which data was manipulated in order to fulfill the needs of the project. This includes the process by which it will be collected, understanding the schema of the photo metadata, creating the basemap imagery and creation of the SpatiaLite database.

4.1 Data Acquisition

Due to the nature of our project being development-oriented, any and all data will not be obtained until deployed offline in Myanmar. Before going out to the field, the smartphone will need the free Photo EXIF Editor app. The process then starts with a community member taking a photograph of a point of interest on a smartphone. Using the location services (GPS) on the phone, the camera metadata is geotagged with coordinate information. The user then adds a text description to photo and saves it on the phone. The user compiles multiple photos with descriptions while in the field. Then, using a USB cable (WiFi and Bluetooth were not tested), the user transfers the photos into a folder on the Raspberry Pi. Next, the user and the library tech (MBAPF personnel) open QGIS and use the ImportPhotos and eVis plugins to create point data and visualize the photos. This new vector data (point) is given reference by adding a raster image basemap(s) to the visualization window. The data can now be appended to the layer in the SpatiaLite database.

Although the data can potentially be added to OpenStreetMap or another volunteered geographic information database if connected to the web, it will live in this offline environment until further called upon.

4.2 Data Acquisition Constraints

We anticipate that data may be constrained physically by the connection to the RPi. In the case of Bluetooth and hardline USB connections, only one transfer can occur at a time. For WiFi, bandwidth is more likely to be the limiting factor in data transference. Additionally, an app may be necessary for this type of transfer although our research was very limited in this area. As mentioned in the system resource requirement requirements, the storage space of the micro SD card (16GB) will be the limiting factor, unless upgrades in storage capacity are being considered. Other considerations might include the familiarity of the community member with the technology itself. Adding a simple text description using the recommended mobile app should be easy to learn, but if a user has never used technology before this may be more difficult than anticipated. Another thing to consider is that the app is in English and the level of familiarity of users with English is unknown.

4.3 Future Data Preparation

It will be very important to monitor the amount of storage used by the anticipated number of total photos and granularity of basemaps and expected geographic extent. By way of the ImportPhotos tool, the vector data lives as a GeoJSON. This file type is very compatible, made to work in many environments such as a web-based GIS. The schema of the SpatiaLite database layer was built to accommodate the photo metadata so we anticipate subsequent styles and brands of Android phones being mostly compatible with any near-future camera apps. Should the schema change, a new layer can easily be built to match these changes. The scalability and compatibility of this setup is its strength. Regarding the QOSM basemaps, there could still be a plugin that works more seamlessly with this setup. In our method, basemaps are individually saved, as opposed to downloading a tiled cache of a specific extent. In an attempt to use this latter method, two other basemap plugins threw errors or caused QGIS to freeze and subsequently shutdown (OpenLayers, QuickMapServices). This said, downloading a tiled cache would be easier and result in one large file. Our method necessities that you choose your extent and scale for each image. While more time consuming, our method could result in fewer unneeded tiles, and subsequently less storage capacity wasted.

4.4 Developing Database Schema Specifications

Table 5 describes the three types of data required for our project. First is the photograph taken by the user (which are pointed to by the *Path* field in the attribute table); second is the point data representing the location of the photograph; and the basemap image, used in visualization after import. Although a simple schema, these components provide valuable information to the map reader.

Table 5. Data Categories

Data category	**Description**	**Filename**	**Source of data**	**Spatial object type**
Photographs	geo-tagged photo	photoname.jpg	community member	Non-spatial image
Photo description	text description	description	community member	Attribute field
Photo points	Brief text description	community _photopoints	GPS(phone)	Point
basemap	Offline basemap tiles	placename_mapscale.jpg	QOSM	Raster image

4.5 SpatiaLite Layer Attribute Table

Table 6 illustrates the physical schema of the SpatiaLite layer, created to accommodate the photo metadata now hosting a text description added in Photo EXIF Editor. The ImportPhotos plugin is used to import the photographs and metadata, resulting in a GeoJSON file. This data is this file is then appended to the SpatiaLite layer. Of note, QGIS listed the width and precision of all fields as "0". After examining the code of the plugin, it appears to be due to most fields being assigned the type "string" rather than an integer.

Table 6. SpatiaLite Layer Attribute Table

Field name	**Description**	**Example**	**Type (data)**	**Width**	**Precision**
ID	Unique identifier	314c7da2-93c8-433f-915f-4ab2a9ce3a4d	String	0	0
Name	File name created by camera	IMG_3738.jpg	String	0	0
Date	YYYY-MM-DD	2018-06-26	Date	0	0

Time	HH:MM:SS	18:41:01	Time	0	0
Description	Description of image	This building is the medical clinic.	String	0	0
Altitude	Elevation in meters	18.0	String	0	0
Lon	Longitude in decimal degrees	96.7987583333	String	0	0
Lat	Latitude in decimal degrees	16.7987583333	String	0	0
North	True (T) or Magnetic North (M)	T	String	0	0
Azimuth	Azimuth of the image taken	116	String	0	0
Path	File path of the photo.	/home/pi/Desktop/IMG_3738.jpg	String	0	0

Next, we look at the workflow implementation.

5. **Workflow Implementation**

This section chronicles the steps our group took to carry out the development work and configuration.

5.1 Logistics

Because we were working in different states with different time zones (PT, MT & ET), it became instantly apparent to our group that we would need to have full control of the unit to fully test it. With the development work fully centered around this unit it, made sense that we each should have one to test individually. We could continually update one another via the Google Hangouts chat platform and use Google Docs to have shared edit access to live documentation. With Bobby owning the same model Raspberry Pi that TASCHA was using, and at a price tag of ~$65, I offered to purchase one of my own. This way we could work in parallel and benefit from each other's advances. TASCHA offered to send us their unit, so that we could eventually configure it and return it with a setup identical to ours.

5.2 Setup

First, it is necessary to set up all the hardware. This includes the Raspberry Pi, Micro SD card, monitor, keyboard, and mouse. Then, from another computer, download the current Raspbian operating system image from RaspberryPi.org. Once you write that image to the Micro SD card (using a program such as Etcher), you now have a bootable Raspberry Pi that you can configure. For GIS software, you would download QGIS 2.14 from Add / Remove Software (which accesses a repository of Raspberry Pi-compatible applications), install the modified ImportPhotos plugin as well as the eVis and QOSM plugins. One could then create a disk image of the functioning Raspberry Pi and make additional clones of the software configuration, or keep it as a backup. Also depicted are the steps of acquiring an Android smartphone and installing Photo Exif Editor or a similar EXIF-editing app. Once these steps are complete, following the first workflow will not require an internet connection, which is one of the goals of the project. The workflow that we developed to accomplish this is depicted below in Figures 6 & 7.

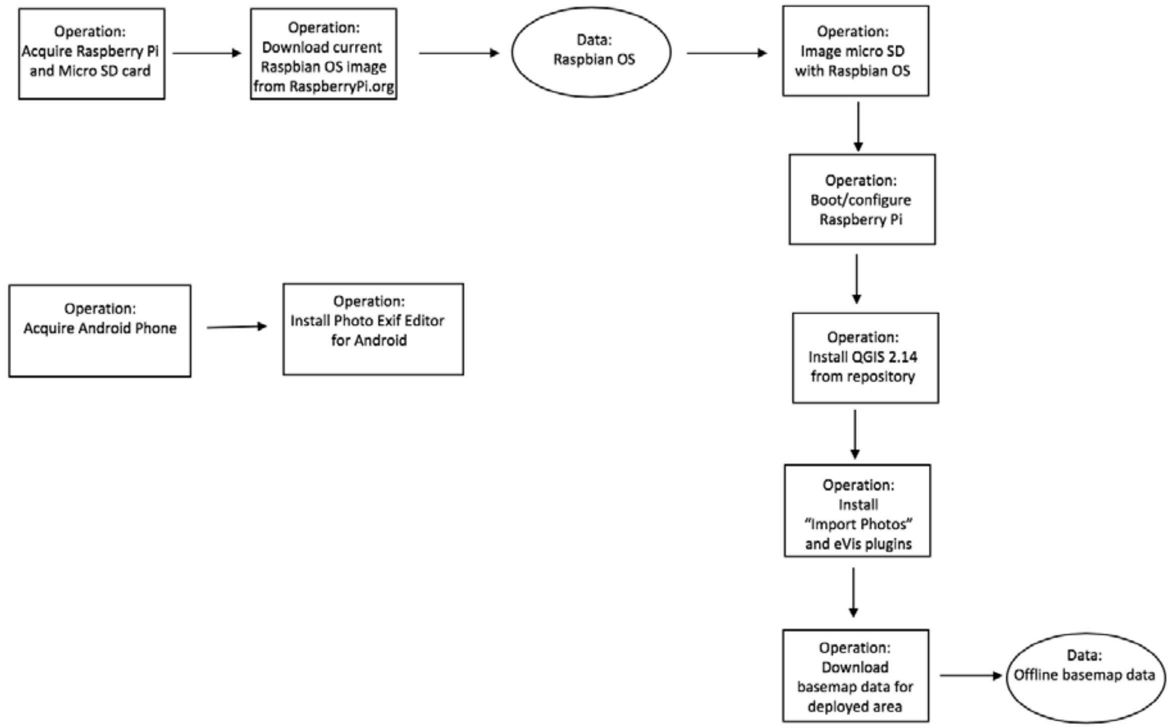

Figure 6. Workflow Diagram for Creating Deployable Raspberry Pi Map Server

5.3 Processing

Any deployment of the RPi setup will need to have a basemap (or set of them) downloaded from the web for use offline. This can be done in QGIS by zooming to an extent, and saving the layer as a geo-referenced image (jpeg, tiff, etc.). If the extent is large (as with a large city or county), zoom levels will need to be taken into account in order to form tiles as basemaps. These will be used to help reference the map reader and for any QAQC that might be done by the user and/or GIS tech/librarian. Next is the creation of a database to host the imported photo metadata. In QGIS, we create a SpatiaLite database, and within this container, we create a SpatiaLite layer. In creating this layer, all field names and data types will be built to accommodate the photo metadata schema.

We are now ready for data collection, documenting occurrences of gender inequality in the form of a geo-referenced photo. This is done on a mobile phone provided by TASCHA. A text description is added to the photo metadata via the mobile app *Photo EXIF Editor* and saved locally on the device. The photos are then transferred to the Raspberry Pi via a USB connection, stored according to a predetermined folder structure (user and date recommended at minimum). That data is then imported using our updated QGIS plugin *ImportPhotos*. The plugin creates a GeoJSON file that hosts vector data. From the GeoJSON attribute table, the data is then copied and pasted into the SpatiaLite layer attribute table. Photos can be visualized via either of the layers since they now share the same file path to the stored photo. This is done by selecting a feature using the *eVis Event ID tool.* The text description is now in a field called Description in both layers as is the other metadata such as date, time and name. Last, we add one of our pre-downloaded georeferenced images to act as a basemap and provide geographic context. This process is depicted in Figure 7. Once this is done, we can make sense of the photo in terms of where the feature exists in the community. Now users can understand where occurrences are and begin to have a conversation about potential trends they might see.

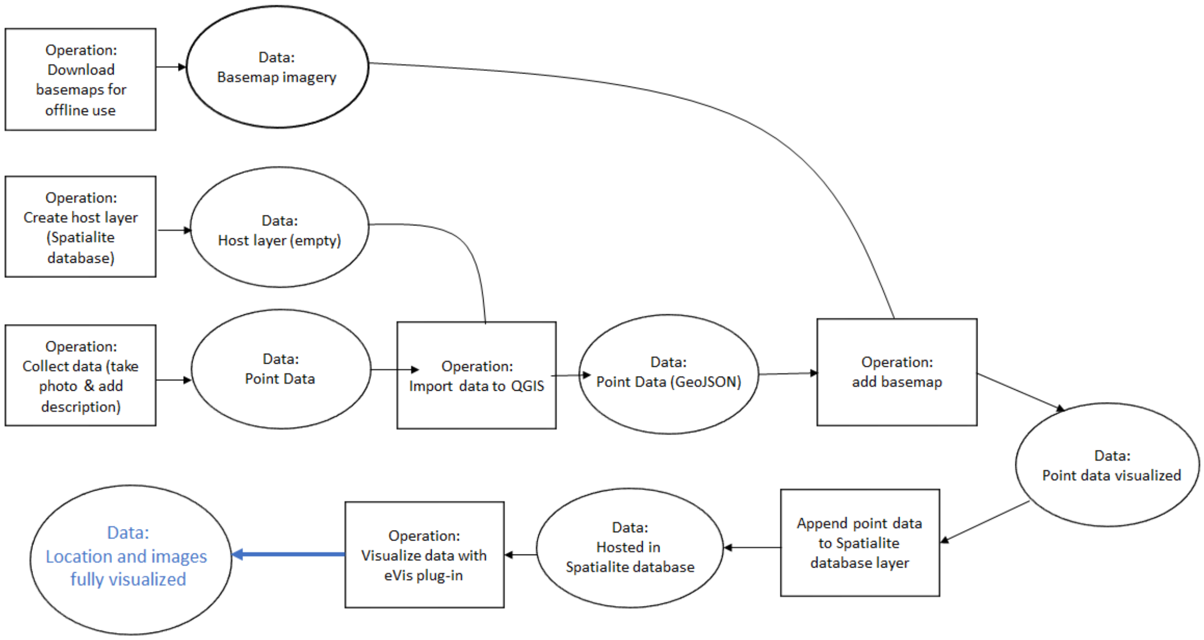

Figure 7. Workflow Diagram for data collection with geo-tagged photos

In the next section we discuss the results of our development work.

6. **Results**

In this section we will summarize what we were able to successfully test and implement, including what didn't work. We will enumerate the major findings of the project by explaining how they were actualized and discuss any struggles and failures. We also provide visual screenshots of the outcomes and tables that further explain our work.

6.1 Need-to-know items

One of the first requests TASCHA had was for us to create a point layer that hosted photo locations, along with their metadata information. Standard categories such as time, date, and name were important to have to better understand and share the photos among the community. This was a success in that we were able to quickly identify a free plugin in *QGIS*, named *ImportPhotos*, that could take a folder of geotagged photos and create a vector data layer in the form of a GeoJSON file. Maybe the most important detail requested in regard to this layer was the additional capability to have a description added by the user, and be brought over in the import process. We were able to accomplish this by editing the open source python code behind the plugin, as described in greater detail in section 2.4. Another crucial component to this function was to identify a way to visualize the photos and description in the GIS. For this we found the plugin eVis, which worked seamlessly in the QGIS environment.

To make sense of the geo-referenced data it was necessary to have a basemap to help orient and reference the map reader. Another plugin was identified for this, QOSM, essentially OpenStreetMaps for QGIS. Initially this worked fine as we were able to take geotagged photos, import then into QGIS, and load a QOSM basemap via the WiFi connection. Our test photos were taken in the U.S., so the basemap was in the U.S. However, when attempting to test the functionality of this see if one could scroll to other continents, the caching lagged considerably. Because of this lag, we decided to look forward to real data collection situations in Myanmar. Would the basemap have fine granularity for a place like Myanmar? In order to test this, we initially edited the coordinates in the image in the folder directory on the Raspberry Pi. Attempting to re-import this image resulted in no change in location, meaning the coordinate date, although visually different in the metadata, had no affected the actual metadata. We then attempted to edit the location data in the mobile app Photo EXIF Editor to that of the Myanmar's capital, Yangon. This re-import did run successfully and when adding a QOSM basemap, was indeed correctly geo-referenced. A check using the same coordinates online in Google Maps supported this successful metadata update. Now saving this basemap image as a .jpg file, created a falsified, yet georeferenced basemap image. Once realizing that we could save any number of these to be deployed in an offline environment, it became a matter of detailing file sizes for specific extents and scales. Depending on the area of interest, and subsequently scale, one could have five to ten of these images, or potentially hundreds (See Table 5). It was at this point that we investigated downloading a tiled basemap cache through the QOSM plugin. In several attempts, we continually got a plugin error. This portion of the research came very late in the overall process and that may have led to the minimal success in this area.

Table 7. Basemap storage estimates

Scale	1:2,500	1:5000	1:10,000	1:25,000	1:50,000
~0.5 sq mi coverage area	61.7 KB		---	---	---
~1 sq mi coverage area	---	84.5 KB	---	---	---
~2 sq mi coverage area	---	---	136.8 KB	---	---
~5 sq mi coverage area	---	---	---	115.0 KB	---
~10 sq mi coverage area	---	---	---	---	120.2 KB

Note: Given the same screen size, scale and extent have an inverse relation.

While maybe not as fundamentally essential, another success came in the import testing. One of the initial questions was whether QGIS would work well for this application. While TASCHA had already identified the software as potentially suitable, our import testing gave us a better understanding of how well the software could process increasing numbers of photos. While the marker was 20 to 100 photos, we tested batches of 1, 20,100, 500 and 1000 (see Table). A single image could be processed in under 1 second, while 1000 photos took just under one minute (See Table 8). In both cases no other software was running and QGIS never stalled, froze, or crashed.

Table 8. Import Speed using QGIS and ImportPhoto plugin

Operating System	1 photo	20 photos	100 photos	500 photos	1000 photos
iOS	<1 sec	1.53 sec	5.71 sec	26.48 sec	54.96 sec
Android	<1 sec	1.73 sec	5.40 sec	27.94 sec	54.17 sec

The other portion of this testing involved recording the memory storage used in these batches as well as in the basemap downloads (see Table 9). Storage of 100 photos used 130-224 GB while 1000 photos used 1.27-2.1 GB. Extent and scale should be carefully considered when storing large amounts of photos in conjunction with a robust collection of basemaps on a standard 16GB memory card.

Table 9. File size estimates

Operating System	1 photo	20 photos	100 photos	500 photos	1000 photos
iOS	1.30 MB	26.1 MB	130 MB	654 MB	1.27 GB
Android	3.2 MB	45.1 MB	223.2 MB	1.1 GB	2.1 GB

The last component of the software development was creating a database to host the data which we did with SpatiaLite, a free and available through QGIS. A SpatiaLite layer was built to accommodate the photo metadata schema (See Table 10) and testing was successful with appending the GeoJSON point features to this new layer.

Table 10. Point of Interest Layer Attributes

Field name	**Description**	**Example**	**Type (data)**	**Width**	**Precision**
ID	Unique identifier	314c7da2-93c8-433f-915f-4ab2a9ce3a4d	String	0	0
Name	File name created by camera	IMG_3738.jpg	String	0	0
Date	YYYY-MM-DD	2018-06-26	Date	0	0
Time	HH:MM:SS	18:41:01	Time	0	0
Description	Description of image	This building is the medical clinic.	String	0	0
Altitude	Elevation in meters	18.0	String	0	0
Lon	Longitude in decimal degrees	96.7987583333	String	0	0
Lat	Latitude in decimal degrees	16.7987583333	String	0	0
North	True (T) or Magnetic North (M)	T	String	0	0
Azimuth	Azimuth of the image taken	116	String	0	0
Path	File path of the photo.	/home/pi/Desktop/IMG_3738.jpg	String	0	0

6.2 Deliverables

6.2.1 Documentation Deliverable

One of the two main deliverables was a process guideline detailing the steps we undertook to successfully implement this GIS setup. That document is the Software Installation Instructions. It includes the following components:

1. Setting up the Raspberry Pi (Raspbian OS installation)
2. Downloading the free open source GIS software QGIS
3. Installing the QGIS plugins
4. Downloading basemaps for offline use
5. Creating a SpatiaLite database to host data
6. Acquiring the mobile app Photo EXIF Editor and collecting data
7. Transferring and importing the photos(data)
8. Appending the data to the SpatiaLite layer and interacting with the data

6.2.2 Configured Raspberry Pi Deliverable

At the onset of the project, TASCHA mentioned a need to have us configure one of their Raspberry Pi units with our prototype development work. This was sent out after our individual units had been configured, as a way of testing the replication potential of our documentation. While duplicating our own Raspberry Pi (by creating a disk image and writing it to a micro SD card) was an option, we decided to go through the steps in our documentation. Doing this allowed us to make minor tweaks to the documentation to ensure readability and eliminate any extraneous steps.

While we expect simple maps to be a central part of the sharing process once data collected, our portion of the development is better visualized with screenshots showing the layers involved, and especially the event visualization tool. Below are examples (Figures 8-10).

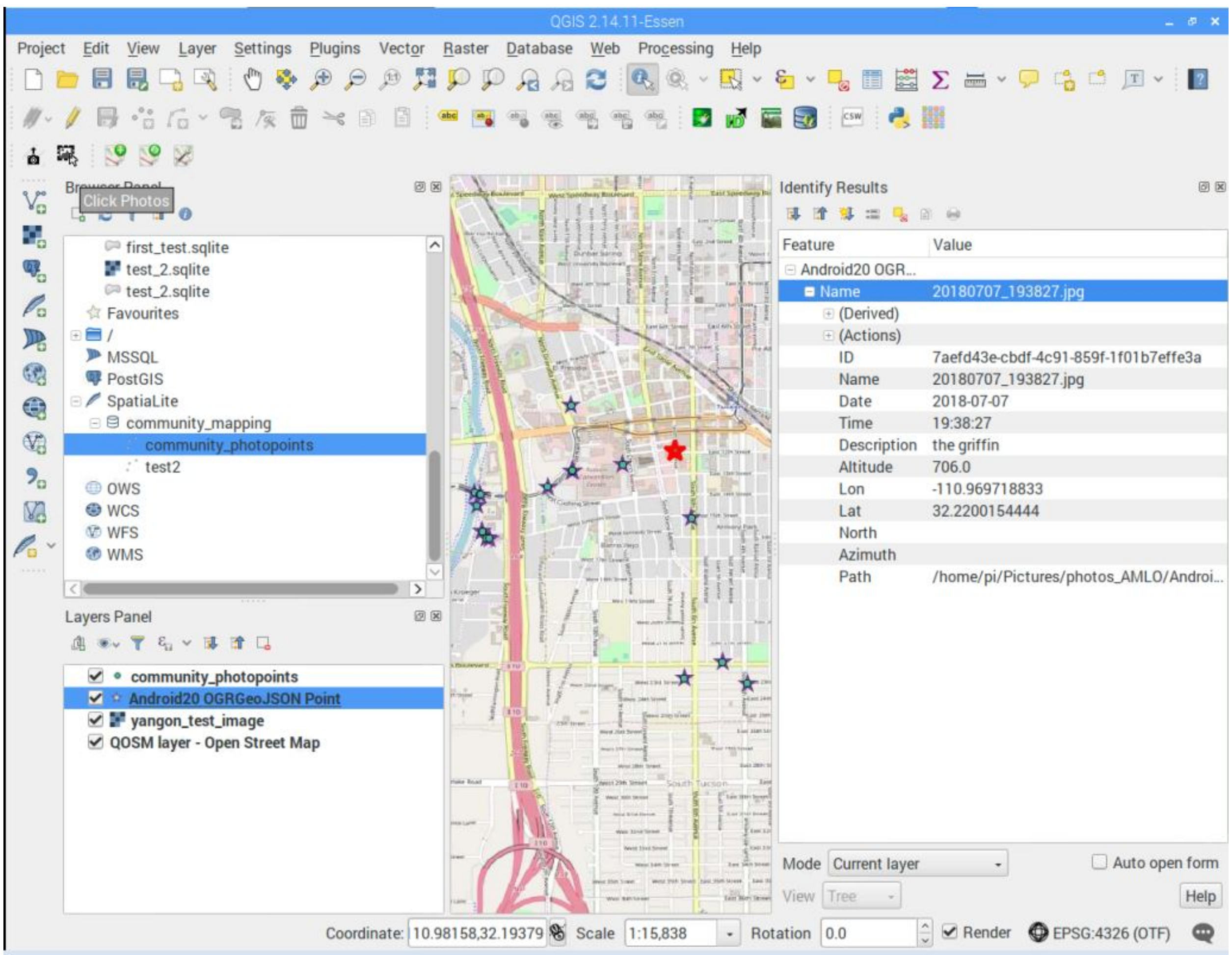

Figure 8. Points imported as a GeoJSON with attribute table (including text description) and basemap.

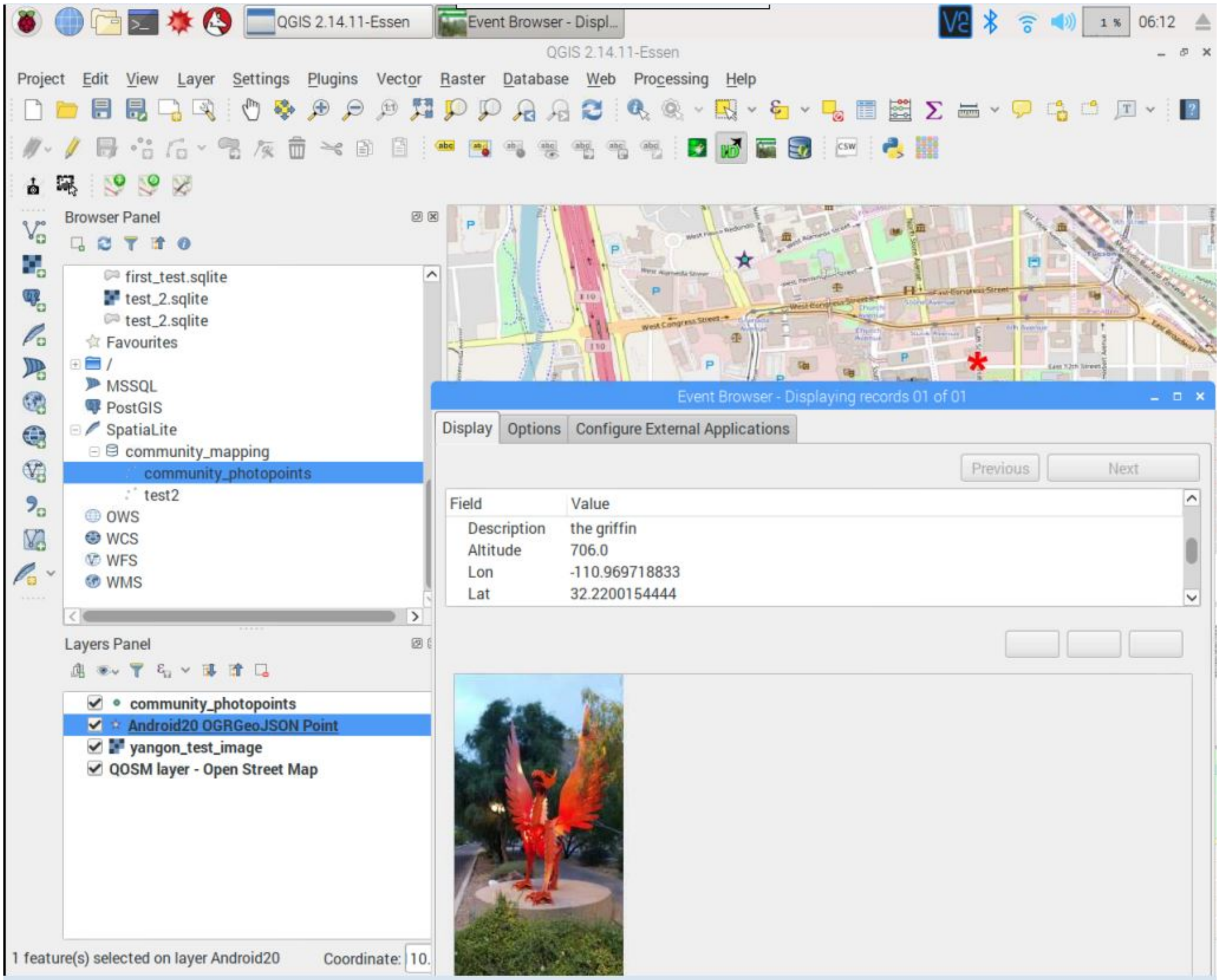

Figure 9. Point with attribute table, basemap and image visualized.

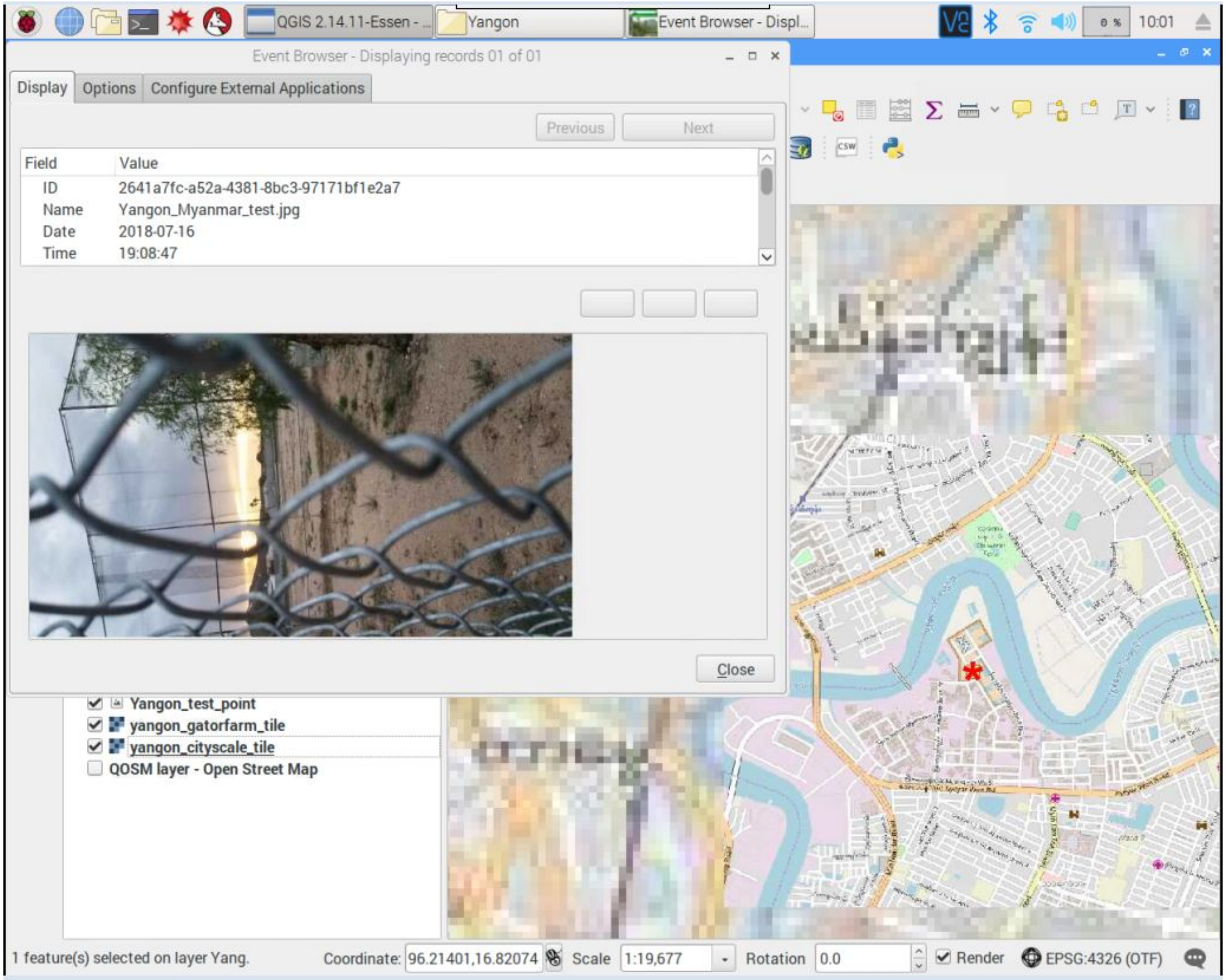

Figure 10. Point with falsified location (Yangon, Myanmar), attribute table, downloaded basemap images (two scales represented) and image visualized.

7. **Conclusions and Recommendations**

7.1 Conclusion

Overall, our development work produced what we think is a successful solution for TASCHA. We were able to get the Raspberry Pi to run QGIS with no issues; created a method for adding text descriptions to photos; identified a free mobile app to pass that metadata along; imported those photos successfully to QGIS; supplied a SpatiaLite database to host the photos; and identified a tool for visualization of the photos with descriptions. In addition, we provided full documentation for replicating the process and tested it by configuring a Raspberry Pi unit for TASCHA that will be returned to them.

Some potential alternatives to our workflow were researched, and while we rejected them as not quite meeting our requirements for this project, they do have a lot of real-world success and a robust developer community. Portable OpenStreetMap (or POSM) was developed by the American Red Cross to bring mapping capabilities into areas without internet connectivity (whether it's due to a disaster or the remoteness of the area). It allows personnel to collect data using mobile apps, and then sync them wirelessly to a PC that can upload data upon returning to an area with connectivity. We reached out to POSM's developers, who stated that their platform does not currently run on the Raspberry Pi (due to the Pi's ARM processing architecture). POSM is part of a larger effort known as Humanitarian OpenStreetMap (HOSM), which utilizes the volunteer OpenStreetMap community to help solve humanitarian problems and develop communities around the world. We think this may still be a path worth exploring in the future, since tools developed for these and other initiatives (such as Field Papers, which allows users to print maps for areas without connectivity, write on them in the field, and then scan them back into the OpenStreetMap database) could prove useful.

7.2 Recommendations

7.2.1 Storage Capacity

One of the recommendations we stress for this project is understanding what the geographic extent needs are for any single Raspberry Pi setup. If the research area is all of Myanmar and only one setup is deployed, accumulation of basemaps at street level granularity will overload the storage capacity of a 16GB memory card. This can be mitigated by determining how many units will be deployed, and determining the granularity needed for each unit. Needs may vary greatly if one unit is assigned to a city area, as opposed to a small village.

Another storage-related issue is the number of photos accumulated on a single unit. Before deployment, the question of how many total photos they anticipate hosting on a single unit. Since basemaps also need to be considered and these are variable as explained, an estimate of maximum photos cannot be provided. Will these units be visited and archived/wiped clean periodically? Would supplying a larger SD card (32GB or 64GB) be useful to mitigate any foreseen storage issues? These answers will emerge as the geographic extent and scale of basemaps for each setup, and temporality of the image lifespan have been determined.

7.2.2 Additional recommendations

There were a few things that we would've done with a longer timeline or more personnel. While the modified QGIS plugin performed the necessary task of taking photo metadata and creating a layer file from it, a more optimal solution would've been developing a plugin (or potentially an application on the Raspberry Pi) from scratch that could replace the multiple plugins our workflow requires. Simply modifying the ImportPhotos plugin took a few weeks to create a stable product that met our needs, and so we did not explore a custom plugin any further. Additionally, we had looked at developing an intuitive Android application that would allow a user to take a photo and write a description (or append a description to an existing photo). After attempting to learn the basics of Android Studio and Microsoft Visual Studio (the latter can be used to create both Android and iOS applications), it was determined that going this route would not be possible during this quarter. While both of these efforts turned out to be beyond our abilities within the time available to us, we recommend that TASCHA explore these options before their field work begins, since they could reduce their dependence on other developers and shorten the learning curve when training volunteers.

Finally, we discovered some of the potential issues with volunteered geographic information (VGI) or public participation GIS (PPGIS). Although our project has taken into account the need for a common schema for data collected (through the use of a SpatiaLite database), human and technical factors associated with spatially and temporally scattered participants could cause unforeseen issues throughout the life of the project. The volunteers and researchers involved in this community mapping project (across different regions) could have different definitions of gender inequality, resulting in semantic inconsistencies. Likewise, despite testing photos taken on several different Android and iOS devices, it's possible that some phones or camera apps may record metadata in a way that is incompatible with our workflow due to a lack of standardization with how manufacturers and developers utilize EXIF tags. Accuracy and reliability have been addressed in previous VGI/PPGIS projects (such as the United States Geological Survey's "Did You Feel It?" program); successful case studies reviewed by the World Bank Global Facility for Disaster Reduction and Recovery reveal several best practices for government and non-governmental organizations to ensure the authoritativeness of VGI/PPGIS (GFDDR 2018). These include verification of data through multiple users submitting information about the same location, not combining existing data with VGI, keeping the public involved during and after the data collection, and ensuring that a "champion" of VGI/PPGIS exists within the organization collecting the data.

8. **References**

Antenucci, John C., et.al. 1991. “Benefits and Costs.” In *GIS: A Guide to the Technology*. New York: Van Nostrand Reinhold, 1991 (ISBN 0-442-00756-6).
Elwood, Sarah. 2002. “GIS use in community planning: a multidimensional analysis of empowerment”. In *Environment and Planning A* 2002, volume 34, pages 905-922.

Global Facility for Disaster Reduction & Recovery (GFDRR). 2018. “Identifying Success Factors in Crowdsourced Geographic Information Use in Government.” Washington, DC: GFDRR. License: Creative Commons Attribution CC BY 3.0.

Huxhold, William, E. 1991. “The Model Urban GIS Project.” In *An Introduction to Urban Geographic Information Systems*. New York, NY. Oxford University Press.

Kojoori-Saatchi, Autoosa Elizabeth. 2015. “The Arab Uprisings: An Assessment of the Roots and Implications of Contemporary Mobilization in the Arab World” PhD diss., University of Nevada, Las Vegas.

Lovelace, David. 2017. “Importing and displaying photos in a QGIS”. Online PDF. http://www.r5r.eu/awir/ImportingGeotaggedPhotosQGIS.pdf

Myanmar Times. 2018. “The real cost of Myanmar’s electricity”. Accessed July 30th, 2018. https://www.mmtimes.com/news/real-cost-myanmars-electricity.html

National Conference of State Legislatures (NCSL). 2017. "’Bathroom Bill’ Legislative Tracking. Accessed July 31st, 2018. http://www.ncsl.org/research/education/-bathroom-bill-legislative-tracking635951130.asp

Raspberry Pi. n.d. “Raspberry Pi specs and Benchmarks.” Accessed July 26, 2018. https://www.raspberrypi.org/magpi/raspberry-pi-specs-benchmarks/

Reuters. 2017. “Seeking end to boycott, North Carolina rescinds transgender bathroom law”. Accessed July 26th.

Technology and Social Change Group (TASCHA). 2018. ”Gates Proposal.” Gates Proposal.

Technology and Social Change Group (TASCHA). 2018. ”Misk Budget Justification.” Young_Misk_Budget Justification 180419 smg. Not public.

Young, Jason C. & Michael P. Gilmore (2013). “The Spatial Politics of Affect and Emotion in Participatory GIS”. In *Annals of the Association of American Geographers*, 103:4, 808-823, DOI: 10.1080/00045608.2012.707596g.

U.S. Bureau of Labor Statistics. 2018. “Average Energy Prices”. Accessed July 30th 2018. https://www.bls.gov/regions/west/news-release/averageenergyprices_seattle.htm

9. Technical Appendices

9.1 Appendix A

Additional Resources

- Initial attempts to identify a QGIS plugin capable of photo importation was aided by the work of David Lovelace (2017).
- TASCHA project lead Jason Young and staff Chris Jowaisas were instrumental in providing feedback and guidance throughout our development process.
- The POSM developers on Twitter (@awesomeposm), who informed us that POSM would not run on an ARM architecture like that used in the Raspberry Pi.
- Marios S. Kyriakou, George A. Christou, and the KIOS Research and Innovation Center of Excellence (KIOS CoE), who developed the ImportPhotos QGIS plugin that our project was heavily reliant on.

9.2 Appendix B

Raspberry Pi 3 B+ Specifications (Raspberry Pi 2018)

- CPU type/speed: quad-core A53 (ARMv8) 64-bit @ 1.4GHz
- RAM: 1GB
- Ethernet: 300Mbps
- Wi-Fi: 2.4GHz and 5GHz 802.11b/g/n/ac
- Bluetooth: Bluetooth 4.2, Bluetooth Low Energy (BLE)
- Storage: 16 GB Micro-SD
- Ports: HDMI, 3.5mm analog audio-video jack, 4x USB 2.0, Ethernet, Camera Serial Interface (CSI), Display Serial Interface (DSI)

9.3 Appendix C

QGIS Requirements

None specified on the qgis.org website. Users on the forum site osgeo.org speculate that this may be because there are rare instances in which it doesn't run, although we couldn't find any reason why this information is unknown. It is supported on multiple platforms including PC, Mac, Linux, Debian/Ubuntu, Android and older systems such as BSD.