Population Distribution Estimation of an Urban Area Using Crowd Sourced Data for Disaster Response

Samuel Lee Toepke
Applications Engineer
Washington, D.C.
samueltoepke@gmail.com

R. Scott Starsman, PhD
Avineon, Inc.
Director, Defense Systems
sstarsman@avineon.com

ABSTRACT

In the event of a disaster, high resolution knowledge of expected population distribution is a boon to the situational awareness of disaster managers and first responders. Knowing the expected locations of large throngs of people can greatly affect distribution of aid and response infrastructure. Effective dissemination of this information can be realized by using a myriad of readily available technologies.

With the modern proliferation of smart phones, pervasive Internet and freely available social media applications, population distribution can be estimated from the constant aggregation of crowd sourced data. Twitter and Instagram both publish geolocated data, which is then processed by a cloud-based, enterprise application to generate heat maps. The heat maps are then shown in a real-time geographic information system that is visible to any mobile device with a web browser.

Keywords

Social media, disaster response, GIS, enterprise application, mobile computing.

INTRODUCTION

Estimating the population in an urban area is a difficult task (Clark, 1951); different cities have different definitions of urban/suburban boundaries, incomplete/erroneous census data might be used, and the results are often dated by the time of publication. Furthermore, the source data does not describe the hourly shifts in transient population that occur throughout the day and week. Even more enticing than low resolution, static population distribution, is an hour-by-hour, block-by-block, always up-to-date estimate.

Current advanced population distribution work includes LandScan from the Oak Ridge National Laboratory (Bhaduri, Bright and Coleman, 2005); which makes population estimates based on multiple inputs, including census information, geographic features and night-time satellite imagery. The resolution of Landscan is about 1 kilometer squared near the equator, and has an ambient as well as day/night population count.

Recent work in increasing temporal and area resolution includes an algorithm based on the disaggregation of census data into residential areas (Aubrecht, Steinnocher and Huber, 2014). Using other modeling algorithms to estimate the ebb and flow of population in business, workplace and industrial areas, a more precise population distribution can be found.

In this paper, the authors describe a culmination of modern technologies that can produce real-time, fine-grained, expected population distribution data of an urban area that is updating hourly, using crowd sourced input. The data is collected using freely available public resources, kept in an online enterprise application, and displayed as a heat map (Wilkinson and Friendly, 2009) using a geographic information system (GIS).

The website for the results can be found here: http://twin-pop.appspot.com/chart.

BACKGROUND

San Jose, CA. is the third largest city in the state of California and the tenth largest in the United States of America (San Jose, California, n.d.). The downtown area contains: San Jose State University (SJSU), restaurants, attractions, and local residences. To gain the necessary population distribution data, the denizens of San Jose were leveraged.

In the year 2015, United States citizens are on their way to using smart devices to implement pervasive computing (Saha and Mukherjee, 2003). Android devices, iPhones, and BlackBerrys are commonplace, with over 50% of American adults claiming ownership of a smart phone (Smith, 2013). These smart phones have powerful processors, sensors, cameras, user interface (UI) devices, accurate location detection, and the ability to always be connected to the Internet.

Twitter and Instagram are two social media platforms that allow users to share personal information about themselves (Twitter, n.d.; Instagram, n.d.). Users of Twitter publish “tweets”, which are 140 character text messages, while an Instagram post allows users to share images and video. Both of these platforms give the user the ability to utilize their device's Global Positioning System sensor to geolocate their post. With a claimed combined user base of over 290 million (Facebook's Instagram says it has 90 million monthly active users, 2013; Wickre, 2013), a deep data set exists over which to glean information. The Twitter API has already been used to investigate the mapping of flood-related events in Germany (Fuchs, Andrienko, Andrienko, Bothe and Stange, 2013), as well as for prediction of user location using a radiation model (Tarasov, Kling and Pozdnoukhov, 2013).

Twitter and Instagram both have a public application programming interface (API), making geographically placed tweets and Instagrams available to compatible programming languages and interfaces. Using the developed enterprise web application, the APIs are queried hourly, and the data is prepared to be used in a heat map.

The two social media platforms were chosen due to their market penetration in the target area, their amenability to location based research, and the ease of access to their APIs. Using other geolocated social media platforms such as Foursquare and/or Yelp would further enrich the data, and is slated for follow-on work.

ARCHITECTURE

This project is implemented as a Java Platform, Enterprise Edition (J2EE) (Java EE at a Glance, n.d.) web application. J2EE enables the developer to utilize technologies such as access control, Java Server Pages (JSP), database access, web services, and batch processes. The code is then compiled to be deployed as an industry standard web application archive file, which can run in many online application servers.

The server chosen for the project was Google App Engine (GAE) (Zahariev, 2009). GAE is a platform-as-a-service application server, is available in the cloud, offers excellent uptime, and contains many convenient APIs.

There are four general workflows continually available to the deployed project. Three are automatic, and one is end user related.

getTwitter()

Using Twitter's representational state transfer API, requests are made to get tweets that are greater than the application's internal most recent tweet. Twenty five requests are made to cover the geographic area of downtown San Jose, CA. The resulting tweets are received in JavaScript Object Notation (JSON), are processed, and then entered into the datastore. When complete, the most recently updated tweet is updated. This job runs once an hour.

getInstagram()

Using web services, this job connects to a .NET server that runs its own processes that retrieve the most recent Instagram posts. An Active Server Pages (ASP) application was designed that subscribed to an Instagram service that reported all Instagram posts within a circle centered on a provided latitude and longitude and a given radius. Once subscribed, Instagram would invoke a given ASP script and provide all new posts for that region in JSON. The results are parsed and inserted into a database. Again, a most recently updated identifier is passed, and posts that are newer are returned in an extensible markup language (XML) format. The posts are entered into the same datastore as the Twitter objects, the most recently updated Instagram post is updated, and this job also runs once an hour.

buildAverages()

It would be too server intensive to generate the UI charts for every user's session. Thus, an averages object was designed, and is updated once an hour with any new tweets/Instagrams that have been processed. The averages object holds the most recently updated values, counters for tweets/Instagrams per day of the week and hour of day, etc.

User Interface

The UI is implemented with JSP, which populates the charts and generates necessary HTML and JavaScript code, viewable with modern web browsers. The site displays two charts, one of the tweets and Instagrams by day of week, and by hour of day. Also displayed is a JavaScript Google Map and an Asynchronous JavaScript and XML interface to query population distribution information by hour of day. The Google Map allows for pan/zoom, street-view, a map or satellite layer, and the ability to view the heat maps by tweets or Instagrams (with different colors), or a heat map with all the data combined.

RESULTS/OBSERVATIONS

The website's heat map gives the end user knowledge as to the population distribution throughout the day and week. Experimenting with the heat map and clicking through different hours shows immediate observations:

- Utilization patterns of academic buildings at SJSU.
- Popularity of businesses.
- Which buildings are dormitories at SJSU and which buildings are neighborhood residences.
- The most highly frequented transit stops.

In Figure 1, one can see an image of the pseudo population distribution of downtown San Jose, CA for Wednesday at 1200. SJSU is hosting classes, while Instagram posts are being produced in the downtown area.

In Figure 2, one can see the same image for an average Friday at 2300. The restaurants/bars of downtown are heavily populated, and the SJSU campus is empty except for the dormitories in the south east corner of campus.

While the heat maps were the primary goal of this investigation, two charts (Figures 3 and 4) were created that show insight into the life patterns of the Twitter and Instagram users: tweets/Instagrams by day of week, and tweets/Instagrams by hour of day.

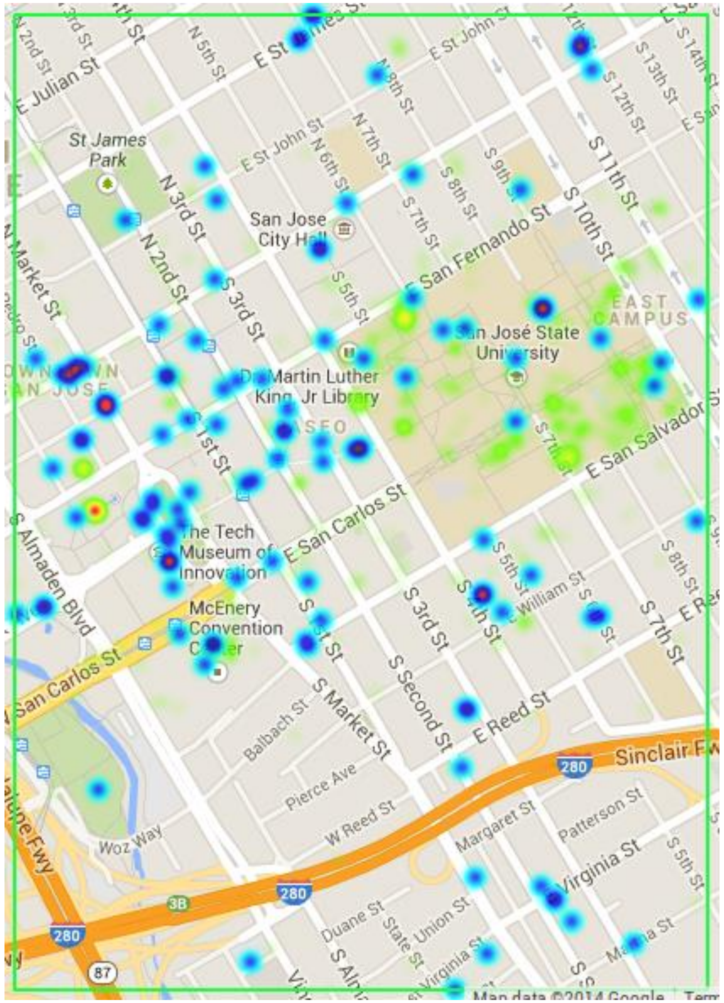


Figure 1: Pseudo population distribution of downtown San Jose, CA., Wednesday 1200. Tweets are green, Instagram posts are blue.

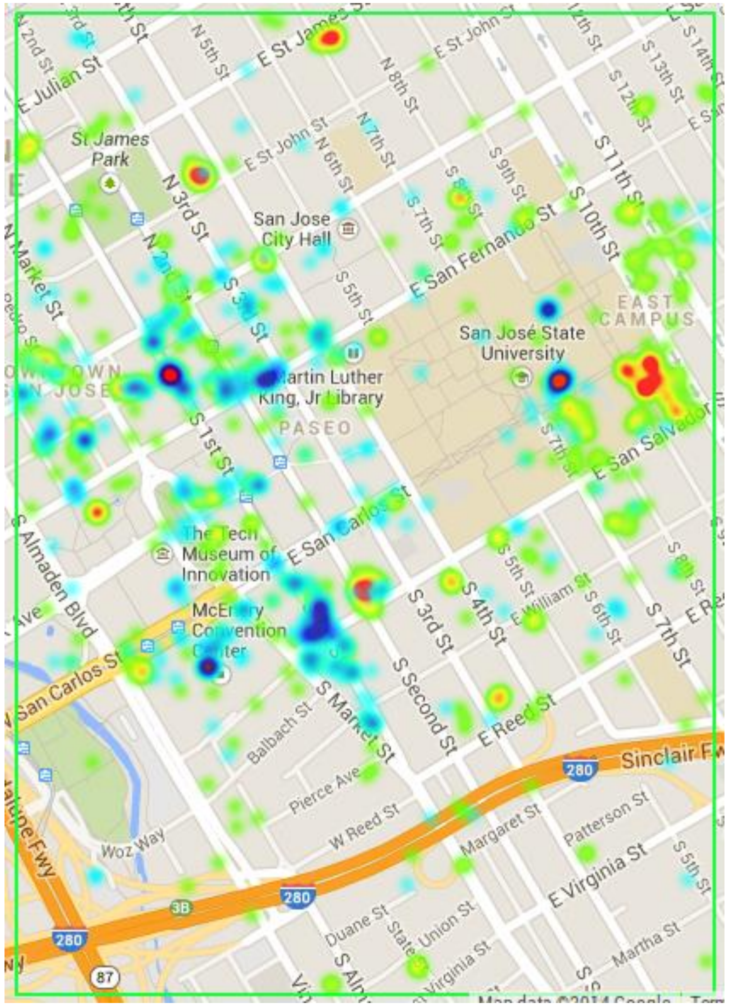


Figure 2: Pseudo population distribution of downtown San Jose, CA., Friday 2300. Tweets are green, Instagram posts are blue.

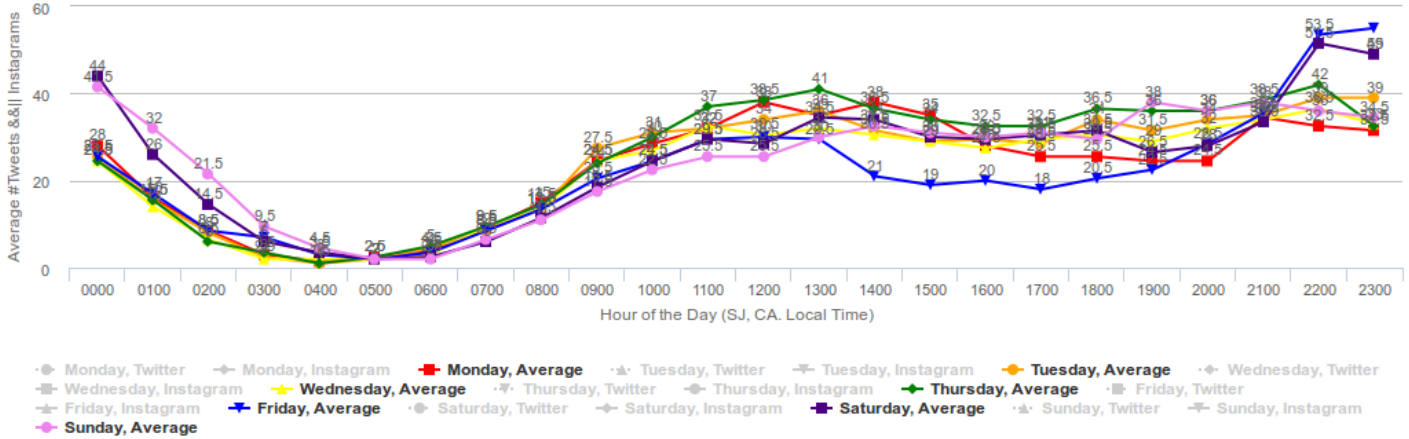
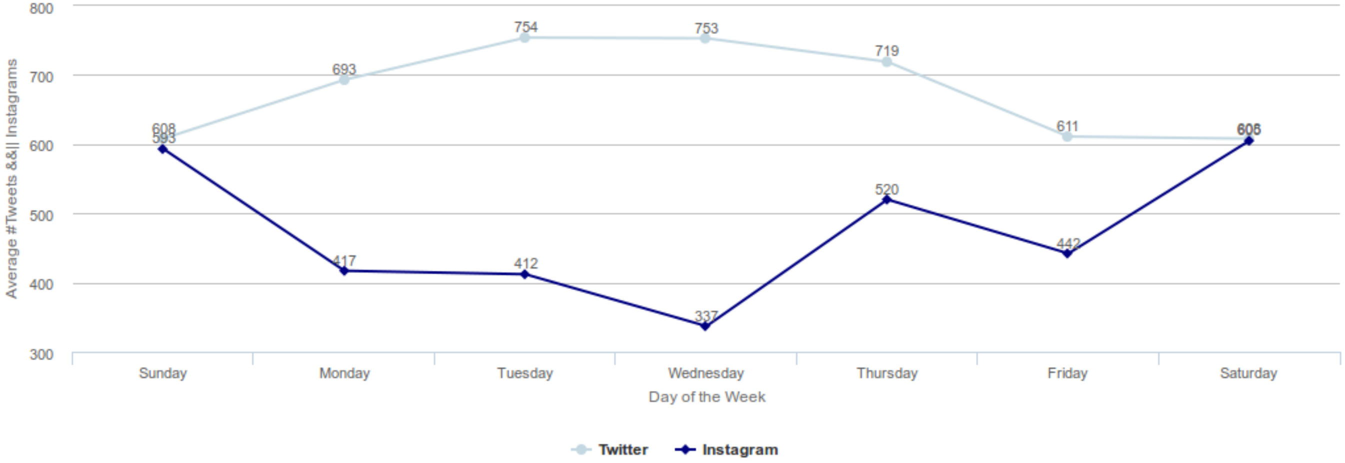


Figure 3 (above): Social media by hour of day. **Figure 4 (below): Social media by day of week.**



There are 21 plots in the `Average Number of Tweets And/Or Instagrams by Hour of the Day' chart; three for each hour (Twitter, Instagram, Average). The chart uses a dynamic JavaScript library that allows the end user to easily enable/disable plots for comparison.

The charts show some immediately interesting results:

- The least social media traffic occurs at ~0500.
- Users tend to tweet during the week days, and post Instagrams during the weekend. During the week, users are in class or at work, and publishing a text message is easier than an image. On the other hand, during the weekend, users are taking part in recreational activities such that taking an image (Franco, 2013) is preferred to typing out a message.
- Saturday and Sunday mornings have much lower social media traffic in the early morning.

Twitter and Instagram's geospatial search APIs have three inputs: latitude, longitude and radius. Ideally, the web application could pass in lat/lon coordinates pertaining to the center of downtown and a twenty mile radius; the returned tweets could then be filtered by a geographic bounding box. Unfortunately, the Twitter search API is limited to returning 100 tweets per request. Also, the requests are limited to 180 requests per fifteen minute window.

Upon experimentation, the authors created an overlapping request grid of 25 requests (see Figure 5), that return an average of ~30 tweets depending on the time of day. This high amount of requests allows a conservative return rate, such that the request isn't saturated in the case a large event causes a massive amount of tweets. The totality of the returned tweets have multiples removed, and are filtered based on a geographic bounding box.

The Instagram API allows a maximum search radius of 5,000 meters. This presents a challenge to the collection of posts over a larger geographic area, as this requires collection of data over multiple, overlapping circles. The Instagram request code allows the creation of overlapping circles but uses the unique image identifier found in the 'Link' field to prevent the storage and counting of duplicate images.

Since each API has limits on the requests per time frame as well as the responses per request, efficiently selecting request parameters is critical.

Using an overlapping circle packing algorithm, as well as work on remote sensing, while modeling each data request as a sensor, the authors are confident a more efficient and/or automated approach can be developed in the future.

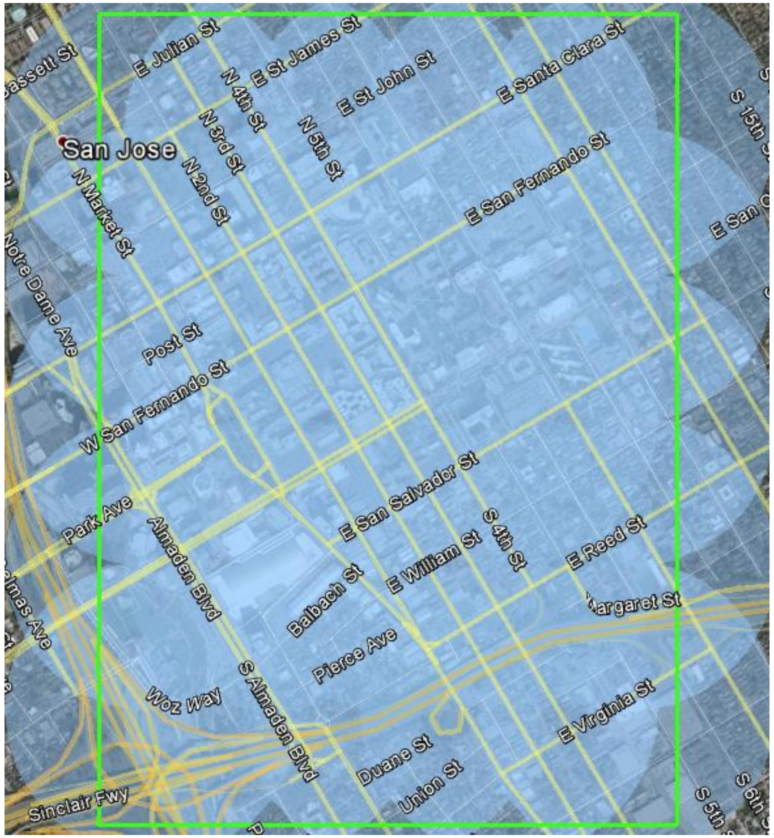


Figure 5: Visual representation of overlapping Twitter requests.

PERTINENCE TO DISASTER RESPONSE

The authors designed this tool based on the use case of an emergency manager, unfamiliar with an urban area, needing immediate, high resolution population estimation to direct aid and response in that area. Using a GIS tool, a responder with local knowledge could easily point out which street corners, which blocks, which buildings, etc. have a large population, based on time of day. An emergency manager at state or federal level will likely not have that domain knowledge, and this GIS tool is meant to supplement their situational awareness.

During an ongoing emergency, the real population distribution will eventually deviate wildly from expectations. This tool will only be useful for a finite amount of time, depending on the emergency. After that initial time period, the fusion of another population estimation algorithm is recommended, ideally seeded from first responder reachback and real-time social media posts (Laituri and Kodrich, 2008).

This prototype was designed to be easily deployed in modern architectures, with the following considerations:

- Cloud based. Using GAE, the cloud benefits of low cost, redundancy, elasticity and scalability are seen. There is no need to deploy and maintain custom hardware. With the current fiscal climate, the cloud allows many local/state/federal/tribal agencies to accomplish more with less. Since the code is developed with J2EE, deployment to FedRAMP (Council, 2012) compliant providers is trivial.
- Modern software paradigms. While currently implemented in J2EE, the software architecture is web services and database oriented, lending to easy deployment in modern enterprise frameworks. If in-house talent for a response authority does not have the necessary skill set, it can be acquired readily. The code base is also readily adaptable to a high throughput Agile software development paradigm.
- Ease of data sharing. Since the social media data is publicly available, there will be a minimum of memorandum of agreement requirements.
- The servlet based data response is amenable to in-place GIS solutions, e.g. Esri ArcGIS, Google Earth, NASA World Wind, etc. The service can be modified to serve data in Open Geospatial Consortium formats, which have already shown benefit in the disaster response realm (Weiser and Zipf, 2007).

With a small amount of cost/setup, this population distribution aggregation and estimation tool can supplement a common operational picture during the first stage of emergency response.

FOLLOW-ON WORK

The work presented here is only a foundation, and leads to many further avenues of investigation and extension.

Data Pool

One of the inherent flaws of this approach is it only takes into account data from those who utilize smart devices and the related social media. This leaves an entire population of elderly, children and technology non-adopters underrepresented. Since the focus of this study occurs in a high urban density area in Silicon Valley, the authors have disregarded the underrepresentation at this preliminary stage.

Future work will explore enhancing the current data pool with those not currently represented.

DynaPop Fusion

Implementing DynaPop (Aubrecht et al., 2014) for the small test area of downtown San Jose, and fusion with the work presented will give a more robust estimation. The aforementioned technologically unrepresented individuals will be mitigated with that data fusion.

Downtown San Jose has a university campus, and many major shopping outlets; thus, the population distribution is subject to changes caused by day of week, whether the university is in session, major holiday shopping, tourist season and other city events. DynaPop's ongoing integration of the Harmonized European Time Use Survey (HETUS) would mitigate these distribution irregularities. Unfortunately, HETUS data is only available from certain participating European countries. Locating a city of similar downtown profile in Europe, and implementing this paper's work for that city would offer exploration of the effects of applying HETUS data.

Heat Map Rate of Change

Locations of rapid population change can indicate incidents of interest, e.g. if a large flux of social media posts in an area is unusual, perhaps an emergency situation is occurring. Using user generated information for abnormal event detection has already been proven feasible by processing social media posts (Chae, Thom, Bosch, Jang, Maciejewski, Ebert and Ertl, 2012); leveraging the average population differential can supplement event detection without undue coding for the end-user. Implementation would include extending the API to request any currently occurring abnormal posting changes, if true, the application would then return lat/lon pairs in those areas.

Data Filtering by User

Currently, the posted data is not filtered by user. A single account could generate many Tweets or Instagrams, artificially affecting the heatmap and making an area look like it has a higher population than in reality. Adding a setting to throttle an account's number of processed posts in a specific time frame can address this.

Another user related concern is cross-posting. A methodology needs to be created to address the possibility that a user has an account on Twitter and Instagram and is posting on the same topic concurrently. Those actions could cause a single user to have effectively double the presence in the heatmaps.

Time Window Experimentation

Currently, each heat map query is done by hour. However, the number of tweets and Instagrams can vary widely within an hour, e.g. a classroom that let out its students at 1615 would be empty for the remainder of the hour. Allowing the API user to specify a time resolution would provide more tightly constrained data.

CONCLUSIONS

A real-time web application that generates an estimated population distribution and presents the data via heat map in a Google Map on a web page has been shown. The application utilizes modern technologies, standards, social media platforms, and programming paradigms. Application to disaster response, observations, and follow-on work has been discussed.

REFERENCES

1. Aubrecht, C., Steinnocher, K., & Huber, H. (2014). DynaPop–Population distribution dynamics as basis for social impact evaluation in crisis management. ISCRAM.
2. Bhaduri, B., Bright, E., & Coleman, P. (2005). Development of a high resolution population dynamics model. Geocomputation.
3. Chae, J., Thom, D., Bosch, H., Jang, Y., Maciejewski, R., Ebert, D. S., & Ertl, T. (2012, October). Spatiotemporal social media analytics for abnormal event detection and examination using seasonal-trend decomposition. In Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on (pp. 143-152). IEEE.
4. Clark, C. (1951). Urban population densities. Journal of the Royal Statistical Society. Series A (General), 114(4), 490-496.
5. Council, I. A. (2012). Federal Risk and Authorization Management Program (FedRAMP).
6. Facebook's Instagram says it has 90 million monthly active users. (2013, January 20). Retrieved January 10, 2015, from http://www.techhive.com/article/2025801/facebooks-instagram-says-it-has-90-million-monthly-active-users.html
7. Franco, James. (2013). The Meanings of the Selfie. The New York Times, 28.
8. Fuchs, G., Andrienko, N., Andrienko, G., Bothe, S., & Stange, H. (2013, November). Tracing the German centennial flood in the stream of tweets: first lessons learned. In Proceedings of the Second ACM SIGSPATIAL International Workshop on Crowdsourced and Volunteered Geographic Information (pp. 31-38). ACM.
9. Instagram. (n.d.). Retrieved January 10, 2015, from

http://www.instagram.com

10. Java EE at a Glance. (n.d.). Retrieved January 10, 2015, from http://www.oracle.com/technetwork/java/javaee/overview/index.html

11. Laituri, M., & Kodrich, K. (2008). On line disaster response community: People as sensors of high magnitude disasters using Internet GIS. Sensors, 8(5), 3037-3055.

12. LandScan - Documentation. (n.d.). Retrieved January 10, 2015, from http://web.ornl.gov/sci/landscan/landscan_documentation.shtml

13. Saha, D., & Mukherjee, A. (2003). Pervasive computing: a paradigm for the 21st century. Computer, 36(3), 25-31.

14. San Jose, California. (n.d.). Retrieved January 10, 2015, from http://en.wikipedia.org/wiki/San_Jose,_California

15. Smith, A. (2013). Smartphone ownership–2013 update. Pew Research Center: Washington DC.

22.

16. Tarasov, A., Kling, F., & Pozdnoukhov, A. (2013, August). Prediction of user location using the radiation model and social check-ins. In Proceedings of the 2nd ACM SIGKDD International Workshop on Urban Computing (p. 8). ACM.

17. Twitter. (n.d.). Retrieved January 10, 2015, from http://www.twitter.com

18. Weiser, A., & Zipf, A. (2007). Web service orchestration of OGC web services for disaster management. In Geomatics Solutions for Disaster Management (pp. 239-254). Springer Berlin Heidelberg.

19. Wickre, K. (2013, March 21). Celebrating #Twitter7 | Twitter Blogs. Retrieved January 10, 2015, from https://blog.twitter.com/2013/celebrating-twitter7

20. Wilkinson, L., & Friendly, M. (2009). The history of the cluster heat map. The American Statistician, 63(2).

21. Zahariev, A. (2009). Google app engine. Helsinki University of Technology.