Citygram One: One Year Later …

Tae Hong Park[1], Michael Musick[1], John Turner[1], Charlie Mydlarz[2], Jun Hee Lee[1], Jaeseong You[1], Luke DuBois[1]

[1] Music and Audio Research Lab (MARL)
The Steinhardt School
New York University
New York, NY 10012 USA

[2] Center for Urban Science and Progress (CUSP)
New York University
1 MetroTech Center, 19th Floor
Brooklyn, NY 11201 USA

{thp1, musick, jmt508, cmydlarz, junheelee, jaeseongyou, luke.dubois}@nyu.edu

ABSTRACT

Citygram is a multidisciplinary project that seeks to measure, stream, archive, analyze, and visualize spatio-temporal soundscapes. The infrastructure is built on a cyber-physical system that captures spatio-acoustic data via deployment of a flexible and scalable sensor network. This paper outlines recent project developments which includes updates on our sensor network comprised of crowd-sourced remote sensing, as well as inexpensive and high quality outdoor remote sensing solutions; development of a number of software tools for analysis, visualization, and development of machine learning; and an updated web-based exploration portal with real-time animation overlays for Google Maps. This paper also includes a summary of technologies and strategies that engage citizen scientist initiatives to measure New York City's spatio-acoustic noise pollution in collaboration with the Center for Urban Science and Progress (CUSP).

1. INTRODUCTION

The Citygram project was initiated in the fall of 2011 and has rapidly developed over the past three years, bringing on board collaborators from the California Institute for the Arts (CalArts) in 2011, New York University's (NYU) Steinhardt in 2012, and NYU CUSP in 2013. The impetus for launching the Citygram [1] [2] project began through observations that current topological mapping systems are typically *static* and focus on visualizing what can already be seen. That is, representing landmarks such as buildings, roads, parking lots, lakes, and other fixed, physical objects. Our environment, however, does not only contain visible objects, but rather, it is filled with a plethora of non-ocular energies that we engage with and are affected by. We continually interact with the environment with our full set of sensory systems throughout the day, week, month, year, and beyond – all senses are non-ocular except one. One such invisible energy source is acoustic energy, which is typically transient in nature and thus difficult to capture via traditional mapping technologies and paradigms. Noticing the absence of what are essentially *soundmaps* – maps that are dynamic and can capture the spatio-temporality of environments – we began to explore technologies and methodologies to capture spatio-acoustic dimensions and overlay them on existing mapping interfaces. The first iteration of this project – *Citygram One* – focuses on capturing the ebb and flow of urban acoustic ecology. This involves the creation of remote sensing solutions; the design and implementation of a server and database capable of handling continuous data streams; the development of urban soundscape taxonomy; and the implementation of the supporting analysis software tools for both research and artistic applications. More recently, beginning in the fall of 2013, our collaboration with NYU CUSP has also led us to pay special attention to measuring, analyzing, defining, and quantifying urban noise pollution as further described in [3].

1.1 Citygram System Overview

Citygram is a large, multi-faceted research project with multiple, focused components that comprise a comprehensive cyber-physical system (CPS). These components include: (1) remote sensing devices (RSDs) to capture urban soundscapes; (2) a scalable server and database schema to facilitate real-time streaming, storage, and retrieval of data from RSDs; (3) a public API and associated software that facilitate access to, and interaction with Citygram for researchers, artists, citizen scientist, and the access to the data; (4) web-based spatio-acoustic exploration tools; (5) soundscape information retrieval (SIR) and analysis components [3]; and (6) a creative sonification component that provides mechanisms for general public engagement with soundscapes.

Citygram (CG) uses a sensor network for capturing, measuring, processing, and transmitting spatio-acoustic data streams that reflect soundscapes in real-time. These features are streamed to our server where they are archived, which is then publically accessible in real-time or as historical acoustic data streams. The concepts, theories and past works behind the CPS, the server architecture, and the database are documented in previous Citygram publications [2], [4]. As can be seen in Figure 1, the current state of the web-based visualization interface facilitates exploration of the sensor network and is accessible by anyone with a web-browser. The webpage provides researchers and the general public with means of discovering RSDs built on the standard Google Maps API interface augmented by CG-specific controls and features. As seen in Figure 2, the visualization webpage also provides users with custom "pop-up" information windows for

each RSD. These pop-up windows allow a user to access details relating to a single RSD, along with a human readable address, current timestamp, associated value for the chosen visualization feature, a playbar for soundscape monitoring, and the unique ID of that RSD.

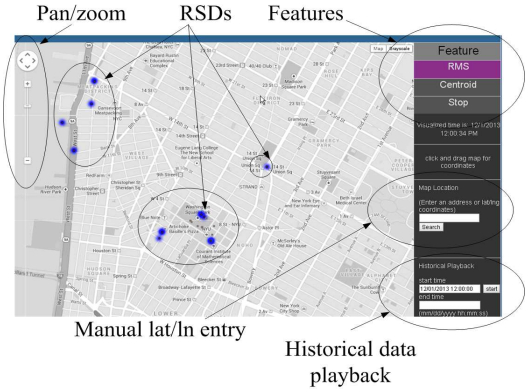

Figure 1. Citygram visualizer showing controls and RSDs in NYC.

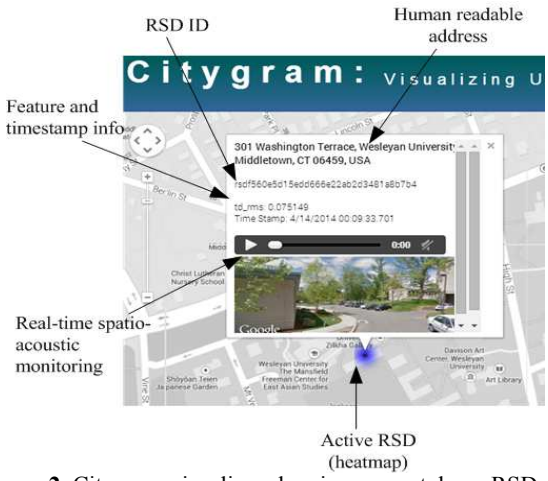

Figure 2. Citygram visualizer showing a smartphone RSD deployed during the 2014 SEAMUS conference at Wesleyan University.

The rest of this paper will present updates including development of fixed-RSD and SuperCollider classes for interacting with Citygram. We will also discuss work related to active deployment of our fixed-RSD network. our paper concludes with a discussion of public outreach and awareness activities accomplished via workshops, open-data access policies, sonification, and compositional possibilties.

1.2 Related Work

One of the earliest works related to environmental sound can be traced to early initiatives led by R. Murray Schafer under the World Soundscape Project (WSP) [5] founded in the late 1960s. Recent projects that are technologically more advanced but espouse many of the ideas found in the WSP, including the notion of soundscapes and soundmaps include *Locustream SoundMap* [6], *Noise Tube* [7], *WideNoise*[1], and BBC's *save our sounds*[2], which are discussed in detail in one of our earlier publications [2]. It is worthwhile pointing out, however, that with the exception of *Locustream SoundMap*, the majority of projects in acoustic ecology have followed, or are still following a model of sound mapping based on audio-snapshot. That is, record at a given time and location; upload to a server; and access recorded historical sounds via web-based technologies. *Locustream SoundMap* is a rare exception in that it is based on real-time audio streaming and monitoring paradigms using an "open-mic" concept where participants deploy the developer-provided custom sensor instruments and share "non-spectacular or non-event based quality of the streams." Audio streams in *Locustream* are broadcast live and accessible via standard browsers.

With the increasing interest in Big Data research a number of projects in the area of urban informatics, urban sound, and noise pollution have also begun to emerge including *radio aporee*[8], *AudioBoo*[9], *de_sense, bruitparif*[10], and *Sensor City*[11]. Both *aporee* and *AudioBoo* also fall into the category of audio snapshot paradigm where *radio aporee* is perhaps one of the most comprehensive ones as it includes non-real-time soundscape capturing efforts via web-interfaces, mobile apps, and an archival mechanism for storing all of its crowd-sourced audio file uploads through the *Internet Archive*[3] online library. *da_sense* employs a "hybrid sensor network" (fixed, wireless, and participative sensing) for urban environmental data acquisition and visualization that includes sound, temperature, brightness, and humidity. For noise measurement, *da_sense* uses smartphones and their custom Android app *NoiseMap* where noise measurements are uploaded by app users and accessed via an online mapping interface in non-real-time. *bruitparif*, initiated in 2005, is another project that focuses on noise in the region of Ile-de-France run by a non-profit private organization with partners from both private and public domains. This project employs a number of mechanisms for its sensor network including "lab vehicles" (essentially cars with microphones), permanent measurement stations, and handheld "sonometers" [10]. Another example is the Netherland-based project *Sensor City*. This project utilizes dedicated fiber-optic networks and high-end audio recording instrumentation for collecting noise data with a focus on exploring human perception and evaluation of urban soundscapes. Using Big Data concepts for its analysis efforts, Sensor City aims to measure soundscapes through the deployment of "hundreds" of sensors to develop automatic sound classification algorithms.

2. REMOTE SENSING DEVICES (RSD)

Citygram's sensor network is based on remote sensing devices (RSDs), which are deployed in a variety of environments, utilizing common computing platforms. Our proposed sensor network design philosophy is based on creating a network that is dense while covering as much space as possible through inexpensive sensor implementations.

[1] http://www.widetag.com/widenoise/

[2] http://www.bbc.co.uk/worldservice/specialreports/saveoursounds.shtml

[3] https://archive.org/details/radio-aporee-maps

The goal is to *densely* populate urban spaces with inexpensive, high quality sensors, as opposed to *sparsely* deploying a few expensive and bulky sensor instruments. In order to meet this design philosophy we developed two types of RSDs: (1) fixed RSDs and (2) crowd-sourced RSDs. Fixed RSDs are calibrated, permanently installed in specific locations, and managed by Citygram. These custom-built RSDs provide consistent, reliable, and calibrated audio related data to our server. Alternatively, our crowd-sourced RSDs are based on the design philosophy whereby any computing device with a microphone and Internet connection can become a Citygram RSD. This includes handheld devices (e.g. smartphones, tablets, "phablets", etc.) as well as desktop and laptop computers running our custom software applications. As further discussed in Section 2.1 the fixed RSDs, are straightforward to assemble and each of the components we use can be purchased on the Internet at low cost. This design and implementation strategy allows for cost-effective, mass deployment of an RSD sensor network by any institution, group, or person wanting to participate in learning more about their surrounding soundscapes.

2.1 Fixed RSDs

Numerous hardware platforms and software solutions were considered and tested during the research phase of developing the Citygram fixed-RSD network. Android-based hardware and software solutions emerged as the clear winner and most appropriate for the needs of our project. In determining our current fixed RSD prototype, we considered various hardware including Alix boards, Raspberry Pi, Arduino, and a number of handheld devices employing the following guiding criteria: (1) audio capture capability, (2) processing power and RAM, (3) low power consumption, (4) flexible OS, (5) onboard storage, (6) wireless connectivity, (7) I/O options/expandability, (8) robustness/sturdiness, (9) low cost, and (10) setup simplicity. The Android mini-PC platform was eventually adopted as it best met the criteria we set in selecting an RSD. The all-in-one solution with built-in microphone, however, proved to be less than ideal with respect to audio recording quality. Although the frequency response of the device's microphone was reasonably flat, we decided to adopt a solution that employed a customized, low-cost microphone further discussed in Section 2.11, and a USB-based audio interface. Figure 3 shows the core components of our current RSD solution that includes the Tronsmart MK908ii Android mini-PC with a A9 Quad Core processor, Quad Core GPU, 2GB RAM, 8GB flash storage, multiple I/O options, including USB, Bluetooth and WiFi at approximately 70USD (as of April 2014) and a USB audio interface.

As detailed in Section 4.1, we are preparing to deploy approximately 50 sensors at Union Square Park in New York City as well as on the California Institute of the Arts campus in Los Angeles. Since our last report we have made significant progress in the selection and development of hardware and software solutions for RSD deployment for both fixed and crowd-sourced mechanisms. This Section provides a summary of our efforts in RSD and sensor network development and its various software and hardware solutions.

Figure 3. Citygram's Android mini-PC, audio interface solution.

2.1.1 Microphone Selection

In selecting an appropriate external microphone for our fixed RSDs, we tested a number of potential microphone candidates including electret, condenser, and Microelectrical Mechanical System (MEMS) microphones. The criterion for selecting our microphone was guided by the following parameters: (1) size, (2) power consumption, (3) frequency response, (4) noise floor, (5) dynamic range, (6) durability, and (7) cost. The microphone that emerged as the best candidate was the MEMS microphone as its audio specifications are high, hardware cost low (less than 2USD each), offers an extremely small footprint (size of a grain of rice when not PCB mounted), and is sturdy. This device is shown in Figure 4 and the

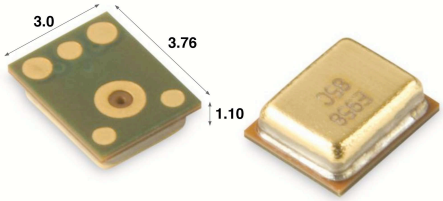

Figure 4. MEMS microphone with dimensions in millimeters.

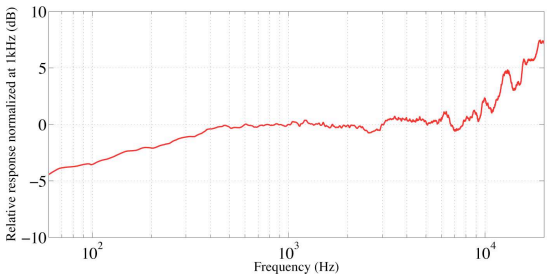

Figure 5. PCB mounted MEMS microphone frequency response normalized at 1kHz.

frequency response of a single MEMS microphone is shown in Figure 5.

The response is favorably flat up to 10kHz, where the rise in sensitivity after this point is a result of Helmholtz resonance created by the microphones inner chamber and PCB port[4]. This results in an increase in perceived sibi-

[4] http://www.edn.com/design/analog/4429422/Acoustic-Design-for-MEMS-Microphones

lance in recordings, which can be filtered out in post processing or on the RSD itself. The MEMS microphone boards that we are currently using for our RSD prototype was built by our collaborators Eric Rosenthal and Michelle Temple from NYU's Interactive Telecommunications Program (ITP).

2.1.2 Outdoor RSD Prototype

To create a viable model for deploying sensors in outdoor environments, a robust and weather-resistant housing solution was devised. The outdoor RSD prototype includes water-resistant housing (15 x 10 x 7 cm^3), Tronsmart mini-PC, MEMS microphone board, USB audio interface and power supply that can be acquired for under $100 total, shown in Figure 6.

As shown in Figure 6, the MEMS board is attached to the bottom of the RSD box and the microphone itself is top mounted on the board with its microphone port pointing towards the ground, through a drilled hole in the housing. The microphone board is mounted on a vibration dampening viscoelastic urethane sheet to reduce structure borne sound and vibration transmitted via the RSD housing[5]. Our fixed outdoor RSDs have been purposely designed to facilitate anyone who is interested in assembling their own, to do so easily at low cost while upholding audio quality integrity. For example, expensive batteries (approximately $52) is not be necessary when an external power supply is available. For other environments when power is only available intermittently such as timer-based lampposts, the added battery component enables 24-hour operation: RSDs use the battery during the day when lampposts are powered down, and charge when lampposts are back online during evening hours. A fully charged battery will allow up to 24-hours of full RSD operation which includes required processing and data streaming.

Figure 6. Complete fixed-RSD (1 – mini PC, 2 – power supply, 3 – USB audio interface. 4 – MEMS mic with board)

2.2 Crowd-sourced RSDs

A number of RSD-type solutions have also been used in projects similar to Citygram where fixed and/or participatory RSD solutions are provided, the latter usually in the form of custom smartphone apps. Our strategies are similar in that we also provide both fixed and crowd-sourced RSD hardware and software solutions. In lieu of the above practices, however, we also provide potential participants with many software solutions to facilitate real-time audio data streaming and interaction with spatio-acoustic data. Anyone with a computing device, microphone, and Internet connection can render their hardware into a Citygram RSD: computing devices can be smartphones, tablets, notebook computers, or desktop computers running our custom software. We are currently focused on publishing the following crowd-sourced RSD software: an Android app, Supercollider classes, Max objects, and MATLAB classes. The Supercollider and Max RSD solutions are described further in Section 2.3. Following the successful release of these packages, we will develop solutions for additional software and hardware platforms, such as Python and iOS.

Active development for a citizen-scientist-oriented suite of smartphone applications has begun, allowing for the acquisition and labeling of sounds, using any Android phone as an RSD. The user experience (UX) for these applications is crucial: users must be encouraged to both record audio as well as accurately label what they've recorded in an entertaining, easy-to-use fashion. Application prototypes allow for users to record sounds, listen to their samples, and perform very simple editing (cropping) of the audio. These samples are then passed to the audio analyzer system to be integrated into Citygram, while a second user experience appears allowing users to categorize rate, and otherwise "tag" their recording. A third screen gives users direct access to the state of the Citygram project, allowing them to see other RSDs on an interactive map. Further research on UX for increasing participatory involvement in the Citygram project will commence this summer in collaboration with the World Science Festival in New York City.

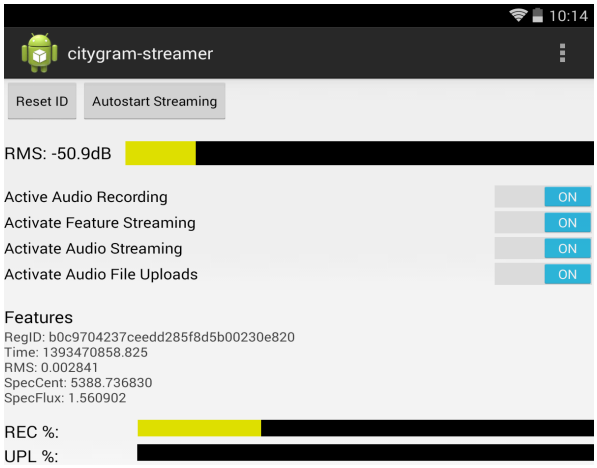

Figure 7. Screenshot of Android streamer app.

2.3 Android Streamer App

The Android Streamer app, shown in Figure 7, is custom software that is installed on our deployed RSDs as well as the Android smartphones of voluntary streamers. This

[5] http://www.sorbothane.com/material-properties.php

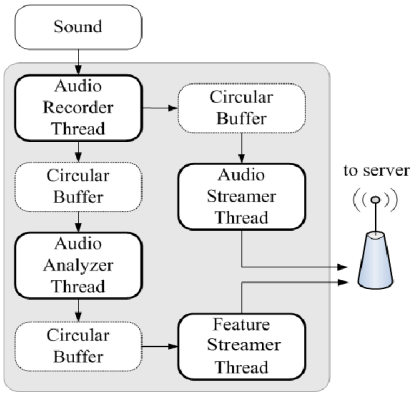

Figure 8. Block diagram of Android streamer app.

app streams both acoustic features and audio data from the soundscape location to the server. Instead of transmitting only the minimal amount of raw data and having the server process everything, we opted for utilizing the client devices for as much processing as possible to reduce the server load. With this distributed computing strategy, our framework will scale with increases in the number of streamers.

2.3.1 Overview

As laid out in Figure 8, the streamer app is structured as four separate threads working in tandem: (1) audio recorder, (2) audio analyzer, (3) feature streamer, and (4) audio streamer, exploiting the multi-core processors in RSDs. The audio recorder thread records the surrounding soundscape using OpenSL ES[4] into two independent circular audio buffers. The audio analyzer thread reads from one buffer, extracts a set of acoustic features, and writes to a circular feature buffer that is passed to the data streamer thread and asynchronously sent to the server via HTTP posts as JSON objects[5]. The other audio buffer is read by the audio streamer thread, which compresses the audio using Ogg Vorbis codec[6] via open-source libsndfile library[7] and streams to the server via Icecast[8].

2.3.2 Audio Blurring

Since streaming audio data can raise privacy concerns, audio data is blurred prior to being compressed and streamed. This renders the resultant processed signal incomprehensible, while the overall "texture" of the soundscape is "preserved." This is accomplished using a modified granular synthesis technique. The basic idea is to segment the audio signal into small "grains" with 50% overlap, shuffle their order, and resynthesize using standard overlap-add techniques. To "preserve" the original soundscape texture, the audio signal is separated into three sub-bands using Butterworth-crossover filters prior to shuffling, such that only the voice signal contained in the middle band is shuffled, while the lower and upper bands remain unaltered. For research purposes, unprocessed audio streams are currently separately compressed with a lossless FLAC codec[9] and asynchronously uploaded to dedicated space in the server. This archive is not accessible by the public, and once the team as identified what features best describe a soundscape, this functionality will be removed.

2.3.3 RSD Auto Update

Once an RSD is deployed on site, it is impractical to gain direct access to the device. Since native automatic updating of Android apps is only supported for the apps that are distributed via Google Play Store[10] an alternative strategy for auto update is employed using a custom "Controller" app. The controller app oversees the lifetime of the streamer and pings the Citygram server with "keep-alive" messages at intervals via HTTP posts. Along with keep-alive signals, the current version of the streamer and the RSD identifier are also sent to the server, and the server responds accordingly, indicating whether any update is available. If an update is needed, the controller downloads the new Android application package file (APK) from the indicated address, closes the streamer, installs the update, and restarts the streamer with the root privileges.

Additionally, both the streamer and the controller have proprietary AutoUpdateAPK software integrated, which allows us to distribute updates over-the-air. This service has a traffic quota for free usage. Although it is not enough for a large-scale deployment, it currently serves in providing as an added fallback for debugging.

2.4 Locative Sonification Tools

One of the creative and artistic ways of using Citygram is in the context of locative sonification and spatio-acoustic telematic performance situations. In the former application area, we have developed a number of custom software programs with a focus on the SuperCollider environment and Cycling '74's Max. A summary of the two software packages follows.

2.4.1 SuperCollider

SuperCollider is an open source, real-time computer music programming language[11] [12] with a wide composer and developer user-base. SuperCollider has been traditional used for composition and sound synthesis but has more recently begun attracting user interest in real-time analysis and music information retrieval applications [13]. As part of Citygram's citizen science software development initiatives, our SuperCollder API currently provides two custom classes for pulling and pushing data from and to our server, with the latter enabling any com-

[4] *http://www.khronos.org/opensles*
[5] *http://www.json.org*
[6] *http://www.vorbis.com*
[7] *http://www.mega-nerd.com/libsndfile*
[8] *http://www.icecast.org*
[9] *http://xiph.org/flac*
[10] *http://play.google.com*
[11] *http://supercollider.sourceforge.net*

puting device running SuperCollider to be transformed into an RSD. The `puller` and `pusher` classes allow for flexible ways to engage in locative sonification, telematic performances, or creating dynamic spatio-acoustic installations via full-duplex data streaming between user RSDs. This solution allows users to access the full functionalities of SuperCollider, as our API provides just another data I/O format within the environment.

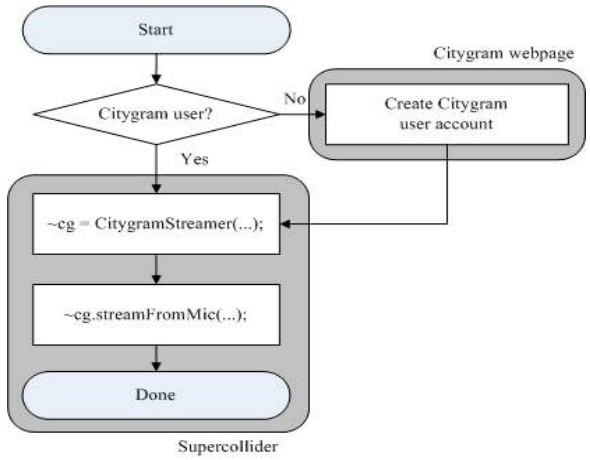

Figure 9. Streaming setup in Supercollider.

The SuperCollider `CitygramStreamer` class, which is accompanied by a well-documented help file, provides methods for users to become *streamers* (a term borrowed from project *Locus Sonus*), transforming the computer into an RSD: streaming low-level feature vectors captured from the user's computer to the Citygram server. Rendering a computer into an RSD is straightforward and simply entails registration on our Citygram server, which in turn creates a unique RSD ID (UID) associated with the particular username and computer ID combination. Once this is established, streaming can begin. As shown in Figure 9, streaming is initiated by creating an instance of `CitygramStreamer` with required arguments username and password, which is followed by calling the object's `streamFromMic` or the `streamFromBus` method to capture audio from the computer's microphone or SuperCollider's internal buses. The internal bus allows for custom signal processing before streaming to the server.

Users can accomplish the location specific placement of RSDs in a number of ways. The most straightforward is to provide valid latitude (lat) and longitude (lng) values during the initialization of a `CitygramStreamer`. Alternatively, if these values are not supplied and this is the first time a user is registering, the computer will assign relative lat/lng values based on the computer-assigned IP address. Finally, the user may choose to manually drag-and-drop their RSD to the "correct" position, through a custom map interface, available within the users account on the Citygram website. The lat/lng is used to display available RSDs on Google maps as shown in Figure 1.

The `CitygramPuller` class is a wrapper for handling API calls and pulls data from the Citygram server to the user's computer. This class allows for pulling single or multiple feature vectors from one or multiple RSDs. RSDs may be accessed in real-time by selecting active RSDs or one may choose to pull past/historic data by providing a timestamp argument to the `CitygramPuller` object followed by the `getData` or `start` method. A convenient way to identify and visualize available RSDs is accomplished via a mapping interface as shown in Figure 2, which is accessible through our public-facing visualizer. A registered user also has a separate interface for viewing and updating personal information and RSD parameters such as lat and lng values accomplished by graphically dragging "drop pins" to a new lat/lng coordinate.

2.4.2 Max

Max is another popular computer music language that is widely used for audio and video data processing [14]. Unlike SuperCollider, Max mainly uses a graphical user interface for real-time interactivity embracing design aesthetics akin to patch-cord-based analog synthesizers. Citygram's Max object is simply called `Citygram~` and can be used for streaming data to, and pulling data from the Citygram server on any platform that runs the Cycling 74' Max software. Similar to SuperCollider, streaming is initiated by providing a username, its associated password, and device ID. As is the case for SuperCollider, audio feature vector extraction is computed directly at the client side and various audio inputs/outputs can be patched to the Citygram object including microphone, line-in, synthesized sounds, or audio files. As in our Android app, the Max version also includes an optional audio broadcasting feature which, when enabled, can be accessed from the Citygram public visualization interface. As is the case for SuperCollider, the Max implementation also provides data pulling options via the `Citygram.puller` method. Pulling from Citygram can be initiated without the need for user registration. Pulling and pushing to and from the Citygram server is achieved via `cURL` commands for Max and Supercollider. The associated pulling `cURL` commands have also been released as a public API, allowing for anyone interested in handling the raw data and subsequent progression through the data.

3. CREATIVE APPLICATIONS

Another area of work that we are developing is using Citygrma in the context of compositional possibilities, distributive-type performances, and sonification. This work is highly influenced by current and past soundscape art practices.

For example, the Citygram team sponsored a sonification contest for the 2014 International Conference for Auditory Display (ICAD). A number of composers submitted installations and concert works utilizing historical soundscape Citygram data as well as real-time sonifications of ad hoc citizen-science RSD networks. Among the pieces submitted, there was a strong interest in comparing the same feature between differing locations. When these locations are related, such as two sensors from the same neighborhood, the resulting music would maintain a similar quality, with slight divergences that offered listeners aural windows into unique events occurring at each loca-

tion. These events would sometimes be mobile (e.g. group of people walking) and this is sonified in the music as the same event causes changes to the musical output in one RSD, followed shortly by the other.

There are also creative efforts currently being undertaken by the Citygram team itself. Jaeseong You has been exploring meta-experience of the data through combinations and compressions of various locations and time-scales into short dance music-like pieces based on Gaussian distributions. This music has captured the hectic nature and machine like regularity of New York City soundscapes.

Building on his previous work in the *Sonic Spaces Project,* Michael Musick has been exploring the use of the Citygram infrastructure for the composition of sonic ecosystems. *Timbral Hauntings*, which is part of the *Sonic Spaces Project* was composed using the same features as the Citygram project, with the goal of incorporating the infrastructure in the next iteration of this piece [15]. This will allow *Timbral Hauntings* to be installed multiple locations, with RSDs feeding feature data between the various systems whereby a single system spanning multiple locations is created forming complex interrelationships between physical spaces, the external agents visiting the spaces, and the digital agents created in the systems software. This will allow for the exploration of meta-connections between interconnected, sonic ecosystems that are directly influenced by the echo's and hauntings of their past.

Not only are the compositional possibilities offered from Citygram rewarding to explore, but they also offer new unique opportunities for soundscape artists. The potential for creation of new spatio-temporal works will perhaps even provide new opportunities for the general public to experience and connect with the characteristics and data from their immediate environment in ways unavailable before. This will hopefully awaken Schaferesque appreciation of environment noise and the composition of urban soundscapes that has been a subject of great interest within the academic murals but has not been yet found equally enthusiastic acceptance by the general population.

4. OTHER DEVELOPMENTS

4.1 Soundscape Analytics

An important component of the Citygram Project is soundscape information retrieval (SIR). That is, using DSP, feature extraction, and machine learning, techniques to automatically classify and detect spatio-temporal acoustic events. In working towards contributing to SIR research, we have begun developing a number of software analysis tools including the MATLAB Sound Analysis Toolbox (SATB) which includes basic DAW-style audio transport functionality synchronized to visualizations (animations); the SATB algorithm extendable plug-in feature; numerous 2D/3D visualization formats synchronized to audio playback; plotting functions that allow efficient and quick plotting of any duration of sound; multimodal and interactive feature space exploration tools; annotation label exploration tools; and a module for pulling data from the Freesound database. Other developments include development of soundscape taxonomy via open-ended *collective listening* techniques and development of baseline acoustic event detection and acoustic event classification tools and algorithms.

4.2 Workshops

In an effort to promote participation and citizen science engagement by researchers and artists we have begun to iteratively develop and present workshops. Our first workshop was held at the 2014 Society for Electro-Acoustic Music in the United States (SEAMUS) conference. The session was divided into a morning and afternoon session, where the first session focused on an overview of the project, its technologies, and concepts. In the afternoon, a hands-on session was held where participants could start using our custom APIs, visualization portal, and create music via spatio-acoustic feature vector streaming. Our second workshop was held at the 2014 ICAD conference, which folded-in lessons learned from our first workshop with improved and additional APIs for SuperCollider and Max.

4.3 Outreach and Collaboration

Our collaborative efforts have quickly grown the past year and now include researchers from CalArts (2011 ~), NYU CUSP (2013 ~), and most recently NYU ITP (2014 ~). Our collaboration with CUSP has been especially interesting due to the synergistic resonance between CUSP's Sound Project and the Citygram Project, which are remarkably similar: CUSP's focus is on urban noise and Citygram (in its first iteration) on acoustic ecology and urban soundscape in general. Our collaboration with CalArts has been in the area of sonification and artistic visualization with immediate plans to deploy 20 RSDs on the Los Angeles campus led by Ajay Kapur. Our engagement with ITP has been one of our most recent collaborative developments where the focus has been on designing and building custom MEMS microphone solutions for our outdoor, fixed RSD prototypes. We anticipate more collaborative opportunities, especially in the realm of using Citygram as a cyber-physical system for exploring different types of soundscapes around the globe including natural wildlife, suburban, and other urban soundscapes.

4.4 RSD Deployment

As briefly mentioned in Section 2.1, we anticipate multiple, mid-scale sensor network deployment by the end of 2014 fall. There is currently only one fully deployed network of sensors at a regional test-site[12] that is being used to actively research and develop software and hardware for the next near-future deployments. Although this site's location cannot currently be disclosed, the sizable number of RSDs at this location are currently being used

[12] Due to legal limitations around this initial deployment test-site, we are unable to disclose this location.

to test and fine-tune forthcoming public sensor network deployment. At the time of this writing, the project is intending to install RSD networks in two public locations in New York City within the next few months. These RSDs will be publicly available and their data will immediately be used for the SIR research.

5. CONCLUSIONS

The Citygram Project remains an active and productive project, encompassing research and development in capturing spatio-temporal non-ocular data; the storage and archival of this data; the analysis and automatic tagging of objects and events within the data; the visualization of no visual tempo-spatial data; and creative compositional and sonification applications of the system. This paper has presented the current state of this complex and intricate project, along with a detailed presentation of many final-candidate decisions that have only been theorized until now. With a widespread deployment about to occur, these latter areas of research are poised to receive a significant amount of attention from the Citygram team (and hopefully the electroacoustic and big data communities) within the coming year.

Acknowledgments

We would like to thank Google Research and NYU-CUSP for their support.

6. REFERENCES

[1] T. H. Park, B. Miller, A. Shrestha, S. Lee, J. Turner, and A. Marse, "Citygram One : Visualizing Urban Acoustic Ecology," in *Proceedings of the Conference on Digital Humanities*, 2012.

[2] T. H. Park, J. Turner, C. Jacoby, A. Marse, M. Musick, A. Kapur, and J. He, "Locative Sonification: Playing The World Through Citygram," in *Proceedings of the 2013 International Computer Music Conference (ICMC)*, 2013.

[3] T. H. Park, J. H. Lee, J. You, M.-J. Yoo, and J. Turner, "Towards Soundscape Information Retrieval (SIR)," in *Proceedings of the International Computer Music Conference Proceedings (ICMC)*, 2014.

[4] T. H. Park, J. Turner, M. Musick, J. H. Lee, C. Jacoby, C. Mydlarz, and J. Salamon, "Sensing Urban Soundscapes," in *Mining Urban Data (MUD) Workshop Proceedings of the EDBT/ICDT 2014 Joint Conference*, 2014.

[5] B. Truax, "The World Soundscape Project," *Simon Fraser University*, 2007. [Online]. Available: http://www.sfu.ca/~truax/wsp.html. [Accessed: 12-Apr-2014].

[6] J. Joy and P. Sinclair, "Networked Music & Soundart Timeline (NMSAT): A Panoramic View of Practices and Techniques Related to Sound Transmission and Distance Listening," *Contemp. Music Rev.*, vol. 28, no. 4–5, pp. 351–361, Aug. 2009.

[7] N. Maisonneuve, M. Stevens, and M. E. Niessen, *Information Technologies in Environmental Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 215–228.

[8] D. Brunow, "Mapping the Sound of the City: Artistic Counter Practice in Hamburg's Regeneration Areas," *Mapping, Mem. CitySchool Archit. Univ. Liverpool 24-26 Febr. 2010 Abstr.*

[9] S. Shilston and J. Lund, "Audioboo: A New Medium For Ubiquitous Microteaching," *in Proceedings of the ICERI2009.*, pp. 1070–1075, 2009.

[10] B. Vincent, V. Gissinger, J. Vallet, F. Mietlicky, P. Champelovier, And S. Carra, "How To Characterize Environmental Noise Closer To People's Expectations," in *Proceedings of Internoise*, 2013.

[11] D. Steele, D. Krijnders, and C. Guatavino, "The Sensor City Initiative: cognitive sensors for soundscape transformations," in *Proceedings of GIS Ostrava 2013: Geoinformatics for City Transformations*, 2013.

[12] J. McCartney, "Rethinking the Computer Music Language: SuperCollider," *Comput. Music J.*, vol. 26, no. 4, pp. 61–68, Dec. 2002.

[13] N. Collins, "SCMIR: A SuperCollider music information retrieval library," in *Proceedings of the International Computer Music Conference 2011*, 2011, no. August, pp. 499–502.

[14] M. Puckette, "Max at Seventeen," *Comput. Music J.*, vol. 26, no. 4, pp. 31–43, 2002.

[15] M. Musick, T. H. Park, "Timbral Hauntings : An Interactive System Re-Interpreting the Present in Echoes of the Past," in *International Computer Music Conference Proceedings (ICMC)*, 2014.