Synthalize: A Crowdsourced Synthesizer

Ahren Alexander
Northwestern University
EECS Department
ahren@u.northwestern.edu

Noah Conley
Northwestern University
EECS Department
noahkconley@gmail.com

Thomas Huang
Northwestern University
EECS Department
thomas.yt.huang@gmail.com

ABSTRACT

The synthesizer is an electronic, and now virtual, musical instrument that can imitate other instruments or generate many new timbres of its own. Since the late 20th century, the synthesizer has grown in popularity and is now one of the most important instruments in the music industry. Modern synthesizers have many adjustable settings that provide for countless combinations of sounds, making them both powerful and versatile. In this work, we describe Synthalize, an interface for rapidly and efficiently choosing a desired synthesizer setting. To use Synthalize, the user searches for and selects a descriptive word that is similar to their desired timbre, and the synthesizer highlights appropriate sounds.

Categories and Subject Descriptors

H.5.2 [**User Interfaces**]: User-centered design; H.5.5 [**Sound and Music Computing**]: Signal analysis, synthesis, and processing

Keywords

Crowdsourcing, human computation, audio engineering, user interfaces, design.

1. INTRODUCTION

Software synthesizers, used by music professionals and hobbyists alike, are prone to follow the model set forth by the physical synthesizers produced by the likes of Robert Moog and David Buchla. The interface of these analog behemoths required hours of familiarizing oneself before meaningful sound synthesis could be achieved by the user. This was the 60s -- why does software today recapitulate this outdated and dizzyingly complicated conceptual model?

The most salient problem with most synthesizers, analog or digital, is the "semantic gap" between the language used by the machine and the language used by the user. While users are apt to describe sounds as "shrill" or "warm," synthesizer systems have no adjustable parameters which directly influence such characteristics of a sound. Instead, users are provided complicated LFOs, envelope generators, and various patch modules to meddle with in attempting to make particular sounds.

This "gap" carries over to the factory presets often installed by the manufacturer as sounds like "DramaQueen" and "Break it Down" populate the synthesizer sound bank [Massive, NI]. When a user is looking for a particular type of sound, these nonsensical names make exploring the synth patch space and finding a setting that matches the user's desired sound very difficult. SynthAssist has looked into this problem and approached it by having users make imitations of the sound they are looking for [1].

In this research we intend to gather several pre-synthesized patches and their acoustic properties, crowdsource descriptions of said sounds, and map these acoustic properties to particular descriptors as determined by the data gathered from those surveyed. For example, if a particular sound is collectively described as "brittle," anytime users search for "brittle" or its synonyms, that sound patch and those with similar acoustic properties will populate the search results.

By establishing relationships between human vocabulary and synthesizer language, users will be able to more quickly or produce the sounds they are looking for.

Figure 1: A standard synthesizer interface and presets

2. GOALS

As music producers, the intellectual interest and practical utility of this project is to both save time (practical) and foster creativity (intellectual) by having the workflow proceed smoothly without being interrupted as often.

Thus, our ultimate goal is to cut down on the time it takes users to find a particular synthesizer patch by constructing an interface to bridge the "semantic gap" and make it possible to select synth presets via appropriate descriptor names rather than complicated settings or random labels (e.g., "DramaQueen"). We intend to do this by crowdsourcing data to tag non sequitur synth patch names with more familiar descriptors. We will use this data to design a unique synthesizer interface: users find a particular sound by looking up words which semantically relate to the sound they are looking for. Upon searching for a semantic descriptor, a list of descriptors will appear, and they can be selected to highlight appropriate sounds resembling the descriptor.

3. PROCESS AND IMPLEMENTATION

In creating Synthalize, we performed the following procedure:

1. Generate 48 synthesized sounds with the Massive synthesizer by Native Instruments. Each sound plays the same melody, but the timbres are different.

2. Gather descriptive words for each of those sounds using Amazon Mechanical Turk. Survey takers (~150) were provided 4 of the 48 sounds and asked to describe each of the provided sounds. This resulted in over 600 descriptor results for the synthesized sounds. This step was derived from the Mechanical Turk procedure used to gather data for Reverbalize [2].

3. Use the results from Mechanical Turk to generate an even larger list of semantic tags for each sound. The semantic tags contain the original descriptors gathered from Mechanical Turk as well as synonyms for each of those descriptors, taken from the Big Huge Thesaurus API (http://words.bighugelabs.com/api.php). This resulted in an overall list of over semantic 1500 tags, many of which describe multiple sounds.

4. Construct an interface to act as a synthesizer controller. Each synth patch is displayed with its original name and can be clicked on to play its associated sound. Additionally, the list of semantic tags can be searched with a search box on the left side of the interface. When a semantic tag is selected, it highlights each of the synthesizer patches that it is linked to.

Tools and resources used to construct Synthalize included Massive (synthesizer), Node.js and AngularJS (creating a web application), and Heroku (hosting the web app).

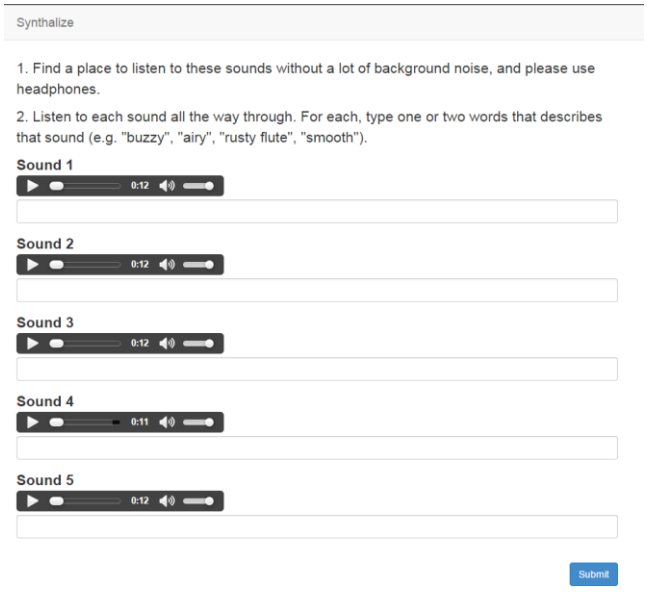

Figure 2: Mechanical Turk survey

4. TESTING AND RESULTS

The distribution of project tasks is as follows:

5. CONCLUSIONS AND FUTURE WORK

Overall, many of our goals were achieved. We were able conduct successful crowdsourcing through Mechanical Turk, and the majority of testers found their desired sounds faster using the tag search than without it.

However, the testing revealed that one of the biggest features Synthalize is lacking right now is size and scope. With only 48 static synth patches currently, a situation occurred semi-often in testing where a user would search for a descriptor and not find any results. This is because the relatively low number of synth patches and their descriptors/synonyms simply did not cover all possible tags and words that could describe the timbre of a sound.

This showed a need for a more dynamic and flexible application. Rather than matching words to prerecorded sounds and displaying those words, the ideal program would instead take a user descriptor and analyze that to adjust a web synthesizer's (such as Web Audio API) settings instead to create a sound in real time resembling the descriptor.

Future work would definitely proceed in that direction, and below is a digital rendering of how a future interface would appear:

Figure 2: Proposed future interface

6. VIDEO LINK

A link to a video describing Synthalize can be found on its webpage at http://synthalize.herokuapp.com/about

References

[1] Cartwright, M. and Pardo, B. SynthAssist: An Audio Synthesizer Programmed with Vocal Imitation. http://music.cs.northwestern.edu/publications/cartwright_pardo_acmmm14.pdf

[2] Seetharaman, P. and Pardo, B. Reverbalize: A Crowdsourced Reverberation Controller. http://music.cs.northwestern.edu/publications/seetharaman_pardo_td_acmmm14.pdf