A Perceptual Assistant to do Sound Equalization

Dale Reed
University of Illinois at Chicago, EECS Dept.
851 S. Morgan St. (M/C 154)
Chicago, IL 60607-7053 USA
(312) 413-9478
reed@uic.edu

ABSTRACT

This paper describes an intelligent interface to assist in the expert perceptual task of sound equalization. This is commonly done by a sound engineer in a recording studio, live concert setting, or in setting up audio systems. The system uses inductive learning to acquire expert skill using nearest neighbor pattern recognition. This skill is then used in a sound equalization expert system, which learns to proficiently adjust the timbres (tonal qualities) of brightness, darkness, and smoothness in a context-dependent fashion. The computer is used as a tool to sense, process, and act in helping the user perform a perceptual task. Adjusting timbres of sound is complicated by the fact that there are non-linear relationships between equalization adjustments and perceived sound quality changes. The developed system shows that the nearest-neighbor context-dependent equalization is rated 68% higher than the set linear average equalization and that it is preferred 81% of the time.

Keywords

Intelligent interfaces, expert systems, learning, perceptual tools, audio equalization

1. INTRODUCTION

Inductive learning can be used to perform an expert skill using nearest neighbor (NN) pattern recognition. This is demonstrated through a sound equalization expert system that learns to proficiently adjust the timbres (tonal qualities) of brightness, darkness, and smoothness in a context-dependent fashion, creating an intelligent computer interface. This is innovative in that it applies the established nearest-neighbor technique to the new application area of performing a skillful perceptual task. This combination has been made possible through advances in computer memory and processor technology, making previously intractable problems now feasible. This work also demonstrates a human-computer interaction (HCI) paradigm where the computer is used as a tool to sense, process, and act in helping the user perform a perceptual task.

The expert system developed here for doing sound equalization is an example of capturing the valuable commodity of human expertise using a computer. Computer learning is needed to help overcome the knowledge acquisition bottleneck for these systems.

Human expertise can be separated into expert knowledge and expert skill. Expert knowledge consists of that which you know how to do, such as knowing when medical symptoms indicate a heart attack or who composed a particular piece of music. Expert skill consists of what you are able to do, such as being able to perform heart bypass surgery or to play a piece of music. Skills as such do not constitute what we think, but rather what we are. An aging professional athlete may still know what to do, but his or her body may no longer be able to execute the action. In our expert system the skill consists of changing tonal qualities (timbres) of sounds through equalization.

In the sections to follow we first discuss the nature of the perceptual task of sound equalization and related work. We then discuss using the computer as a tool to capture expertise (section 2). In section 3 we look at the underlying computational approach used, that of Nearest Neighbor Inductive Inference. Then we discuss setting up the experiment using equalization to change timbers of sound (section 4), with conclusions presented in section 5.

1.1 A Perceptual Task

We define a perceptual task as a task where sensory input is processed to appropriately perform some action, e.g. riding a bike or vocal harmonization. We differentiate between *sensing* and *perceiving* in that perceiving takes the additional step of incorporating the sensory input into some sort of usable representation. Perceiving is not just observing, but additionally apprehending. We also differentiate between an "ordinary" perceptual task and an expert, or skillful, perceptual task. Many people can drive a car, but few have the skill to drive in a race. Many people can tell which of two equalizations for a piece of music they prefer, but few have the skill to isolate which frequency bands cause the differences.

Though early Artificial Intelligence (AI) researchers felt sensory-rich mundane tasks such as vision or locomotion would be easier to solve than "expert" tasks such as medical diagnosis, the opposite has proven to be true. We show how a computer system can be used as a tool to aid the user in both perceiving and performing a sensory-rich task, giving a non-expert an expert level of performance.

1.2 Sound Equalization

Sound equalization is used in public address systems, recording studios, movie theatres, and stereo systems. At a very basic level it is encountered on home stereo systems as the treble and bass tone controls. These act as amplifiers and filters changing the amount of energy in different frequency bands. Equalization is used to make a sound be perceived as more natural sounding, since audio equipment and room acoustics change aspects of the original sound. Secondly equalization is used to give a sound a new property, such as making drums sound more resonant or removing a harsh "nasal" quality of a singer's voice. Bartlett [1] has a description of terms commonly used to describe timbral qualities.

Typically when a sound engineer is setting up a sound system, the system as a whole is first equalized to compensate for the equipment and the listening environment. Next individual channels are equalized for the microphones on particular instruments or other sound sources. Expert sound engineers are those who have developed through experience the ability to hear a sound and isolate exactly which one or several frequencies (out of 31 possible bands) need to be changed to give a desired effect. This is complicated by the context-dependent nature of equalization.

1.3 Related Work

Our Human-Computer Interaction (HCI) paradigm of using the computer as a perceptive tool is related to computer representations of sensory data used to create virtual environments. For instance visual, aural and tactile feedback are used by biochemists in the pharmaceutical drug design process through a simulation representing the atomic interaction between molecules [5]. Users get tactile feedback as they manipulate the image of a molecule they are building. In this case the computer is used as a sensory tool in the virtual environment, however it isn't an intelligent tool in that it doesn't learn. Other examples of virtual environments are three-dimensional computer games, micro-surgery, and remote robotic control. Processing of sensory data is also used for autonomous vehicle navigation [12] and speech recognition [9]. The virtual environments described above are used to present sensory data, though the interface is not used interactively to enhance a user's skill level.

There are two notable examples where a computer does learn to perform skillfully. The first is Lee Spector's GenBebop program [10] where a genetic algorithm is used to create improvisation based on a short underlying musical segment. The result is very interesting, though arguably not expert performance. Second is Harold Cohen's AARON system [2][2] that automatically generates paintings through the use of an elaborate rule-based system with a flat-bed plotter. Our work differs in that the computer is used as a tool to aid the user in both perceiving and performing an expert task.

2. USING THE COMPUTER AS A TOOL

In order to use the computer as a perceptual tool, the user must be an integral part of the system. This exploits both the memory and processing power of the machine as well as the intuitive and synthesizing ability of the user. Both the machine and the user perceive and remember independently of each other, but productive synergy can arise when they are combined.

2.1 General Schema to Capture Expertise

Consider the schema used to capture expertise shown in **Figure 1**, applied in this work to the expertise developed by a sound engineer. Our goal is to *externalize* a sound engineer's *internal* expertise, capturing it in a form which can be reused by a non-expert.

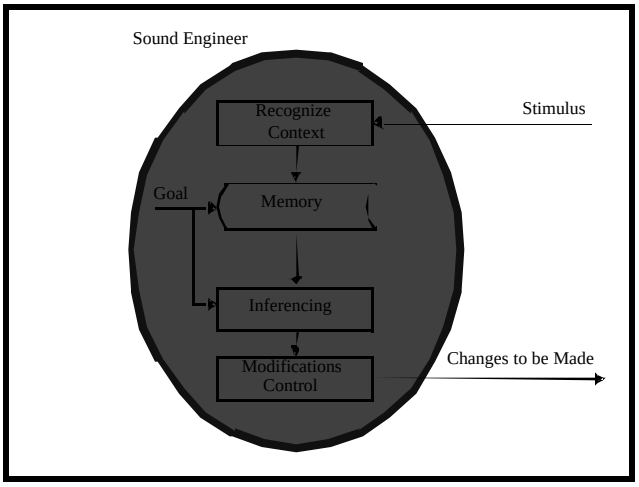

Figure 1: Schema used to capture expertise.

The engineer first recognizes the features or context of the present sound, then remembers similar sounds and equalization changes made in the past with respect to the desired outcome. This information is used to infer similar equalization changes to be made in the present case. Our intent is that this process be externalized to the point that a user can think only about the goals and need not have the expertise to match features or infer equalization changes. The same schematic would apply to perceptual tasks other than our example of sound engineering. By introducing a computer into the loop, the stimulus, context, goal, and resulting

changes can all be remembered for later use, possibly by a non-expert.

2.2 Capturing Expertise Using a Computer in the Loop

Figure 2 illustrates how the expertise-gathering schematic from **Figure 1** can be implemented in a computer system. The system must first be trained, accumulating the body of experience that constitutes the system's expertise. The second phase, performance, uses the accumulated knowledge using inductive inference as shown by the thick light-gray lines. In order to train the system an expert user perceives the stimulus and is given a goal. The user manipulates the stimulus using the computer to achieve an aesthetically pleasing difference with respect to the goal. The context of the original stimulus (the auditory identifying signature), along with the new computer changes for the selected goal are then stored in the database.

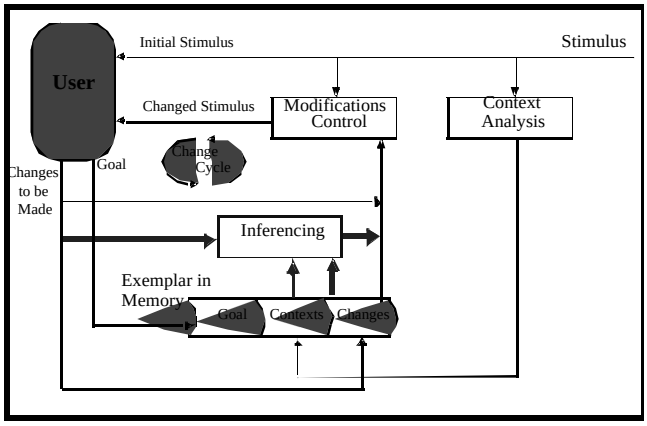

Figure 2: Capturing expertise with a computer in the loop

In our application to equalization the Stimulus is a sound. The Context Analysis yields a representative "signature" made up of a measurement of the average energy per frequency in each sound. The Goals are changes in the timbres of brightness, darkness, and smoothness. Each example's Changes are the equalizer settings used to implement the goal, and the Modifications Control is an audio equalizer.

For the performance phase, we add the inferencing module. As before, a stimulus (e.g. a sound being played) enters the system, but this time the user selects a goal. The system does pattern matching on the stimulus' signature, finding the *n* most similar previously recorded examples (signature-goal pairs) in the data base, using nearest-neighbor pattern matching. The system then makes the same (or very similar) changes to the present stimulus as was made to the previously captured stimuli (nearest neighbors) for the same goal. The user can provide corrective feedback, with these changes added to the database as a new example. Note how the computer is used as a tool to help perceive the input (Context Analysis), induce the proper action to be taken (Inferencing), and also cause the resulting perceptual change (Modifications Control). The system also has the ability to change dynamically according to user preferences by remembering the users' feedback in cases where the suggested change was inadequate.

Now let us take a look in more detail at the Inferencing module as implemented here using nearest neighbor pattern recognition.

3. NEAREST NEIGHBOR INDUCTIVE INFERENCE

Symbolic Artificial Intelligence processing is involved with "figuring out the rules," or coming up with the underlying primitives and their relation to each other. Sometimes it is not possible to figure out the rules or in fact not necessary, particularly in cases where you are capturing a *skill* rather than knowledge (e.g. riding a bicycle.) In this research we are using knowledge without completely understanding its primitives and their relationships. Rather than "figuring out" or reasoning, we use pattern matching to inductively solve new problems in analogous ways to previously seen similar situations - an "expertise oracle," as it were. Stated another way, "Intelligence is as intelligence does." This is done algorithmically using Nearest Neighbor inductive inference.

3.1 Related Inductive Methods

Genetic algorithms [4] are one of the most popular inductive inference methods, though they suffer from lengthy training time. Modified classifier systems [3] reduce the training time, but have great sensitivity used in reward and punishment values. Decision trees [8] provide efficient lookup, but suffer from a need for very large data sets and a length set up time compared to the Nearest Neighbor (NN) [3] approach. Although knowledge of the relationships between examples is more opaque when using NN, it has the advantages of being very straightforward, sensitive to local populations, and adaptable to dynamic changes in the data.

3.2 Nearest Neighbor Description

Nearest neighbor is an example-based pattern recognition approach where all the data points are stored in an n-dimensional space (hypersphere). A new example is mapped into that space and its predicted outcome is computed from the outcomes of it neighbors, that is the points close to it. These points share similar characteristics.

Consider applying NN to credit-risk analysis, where information from a credit card application is evaluated in order to assign a credit rating to an applicant. Applicants whose credit rating falls below some threshold will not be given a credit card due to the risk involved.

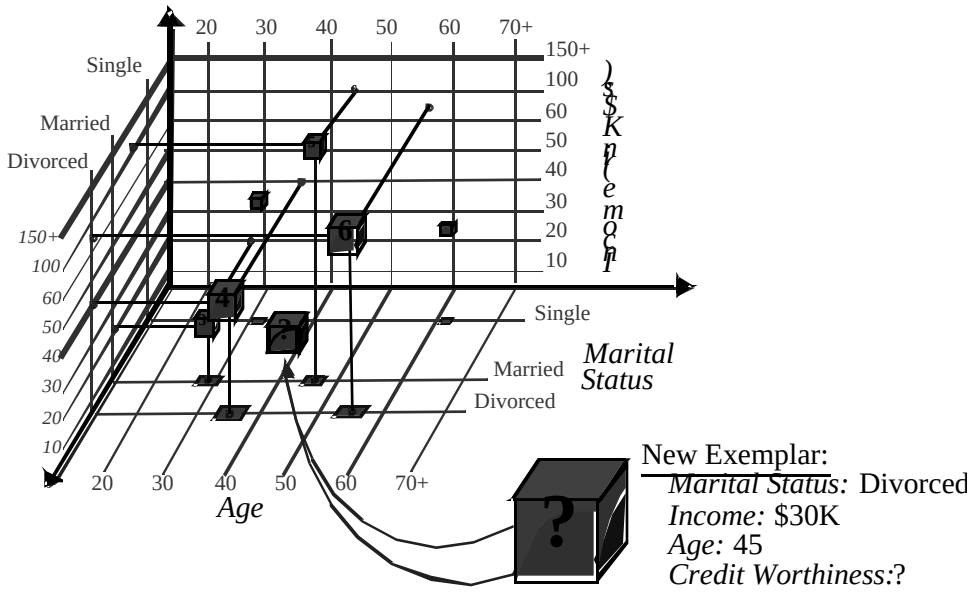

Figure 3: Placing a new example by its nearest neighbors. Outcome of credit-worthiness is determined by outcomes of its neighbors.

This is illustrated in **Figure 3,** where we are trying to predict the credit-worthiness of a loan applicant based on marital status, income, and age. The outcome of credit-worthiness is represented by the numbers inside the boxes, and the location of the boxes reflect the other fields' values. We have chosen credit-worthiness to be scaled between 1 and 10 for this example, where 10 is most credit-worthy. Placing the new example into the hypersphere of only 3 dimensions in this case we find that it is closest to two other points whose outcomes are respectively 4 and 6. The new example's outcome is then some function of those values, either by some sort of weighted average ("5" in this example) or the value that occurs most frequently in cases when there are multiple "close" values.

In our implementation applied to sound equalization, each field (or dimension) is actually a measure of energy in one of the frequency bands averaged over the length of the sound. Experts' use of the system serves to train it, populating the NN search space. When a non-expert uses the system, new sounds are compared to existing ones, and new changes made are similar to those done in the past for similar sounds the system has already heard.

4. CHANGING TIMBRE USING EQUALIZATION

The goal of the implementation was to create a trained system usable as a tool by a non-expert to do expert sound equalization (eq), changing the tonal quality, or timbre of a sound using equalization through an implementation of a NN inductive inference system. We discovered that context needs to be taken into account in affecting timbre through equalization. In other words you can't just always "do the same thing" to give a desired perceptual effect. It depends on what the underlying sound is.

4.1 Adjusting Timbres

We looked specifically at the timbres of brightness, darkness, and smoothness, as illustrated in **Figure 4** Brightness can be thought of as high-frequency emphasis, with weaker low frequencies. Darkness can be thought of as the opposite of brightness, with lower frequency emphasis and a decrease in high frequency energy. A sound is smooth if it is easy on the ears, not harsh, with a flat frequency response, especially in the mid-range, with an absence of peaks and dips in the response.

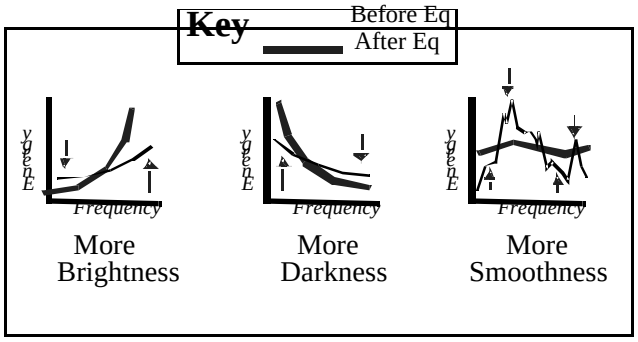

Figure 4: Equalization changes for three timbres.

As mentioned previously, what makes this equalization task difficult is that the equalization changes are context dependent. What makes one sound brighter may not work for another. Making a cymbal brighter would involve increasing the energy in the highest frequencies available (the sliders furthest to the right on a graphic equalizer), but doing the same thing to an electric bass sound may not make any difference at all. This is because there is no energy present at those high frequencies to begin with. When adjusting sliders to make an equalization change, one must take into account the characteristics of the underlying sound. It isn't possible to just always do the same thing to every sound for a desired effect. Equalizations are not only context-dependent, but they are non-linear as well. Moving certain sliders could make a sound increasingly smooth, but after a point continuing to move the same sliders in the same direction could give an unpleasant quality to the sound.

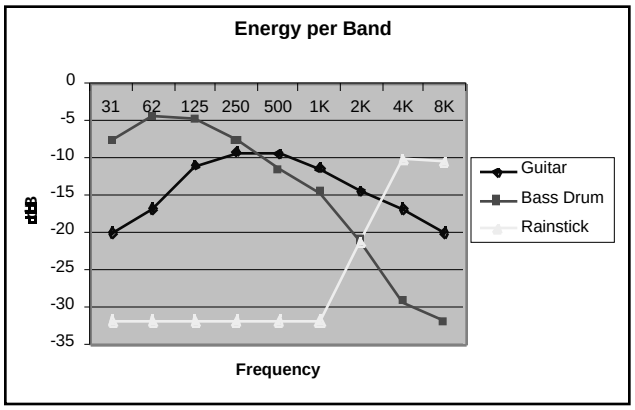

Figure 5: Energy per frequency band for three sounds.

For example, consider the goal of an increase in brightness applied to the three sounds (bass, acoustic guitar, and rainstick) whose energy graphs are shown in **Figure 6**. To make the bass sound brighter we would want to increase the energy in the bands 500, 1K, and 2K. To make the rainstick sound brighter, however, we would have to increase the energy in bands 4K, and 8K, which is different.

4.2 Experiment Setup

The 17 subjects used in training and testing the system were sound reinforcement professionals as well as some music students. Subjects were first given a hearing test to determine their ability to hear the difference between different equalizations. Next the system was trained as users performed equalizations using the system. Then users evaluated the system's level of acquired expertise. Each of these 3 phases (Hearing Test, Training, Testing) is further described below.

The physical listening chamber was lined with sound baffling panels setup to eliminate any early reflections. The graphical user interface consisted of a 10-band[1] on-screen equalizer with real-time measurement of energy per frequency band. The sounds used were taken from unprocessed studio master tracks of typical folk/rock music (e.g. vocals, guitars, basses, drums, etc.). 41 stereo sound segments approximately 15 seconds long each were used for the training session. The testing session sounds were a distinct set of 10 more sounds. In order to be able to do pattern matching a "signature" consisting of measurement of energy per each of the nine frequency bands over all 15 seconds was taken for each sound, with a filter to exclude quiet spots in the sound segment in the averaging. For example, we did not want the measurement of average energy in a drum sound to include the silences between beats. The signature of energy in the nine bands was used to place each sound in a nine-dimensional space (nine dimensional array) for searching using nearest neighbor.

4.3 Hearing Test

Before training the system we ran a brief hearing test. Users were presented with two sounds, where one of them sometimes had an equalization change applied to it. He or she then indicated whether or not the two sounded the same or different. 30 such judgements were gathered, giving an indication as to how well the user could discern equalization changes. Results ranged from 60% to 90% correct judgements for the 17 subjects.

4.4 Training Phase

The interface shown in **Figure 6** was used by each of the 17 subjects in equalizing the 41 sound segments. All examples with "Brightness" as a goal were done first, then those for "Darkness" and finally "Smoothness." The system first highlighted the desired goal, where the goal was to give an aesthetically pleasing increase in brightness, darkness, or smoothness. The user then made equalization changes using the on-screen eq sliders as the sound was playing, trying to achieve the highlighted goal. The "Flat Eq" and "Changed Eq" buttons allowed subjects to compare the changed sounds to the original sounds. These controls were used in real time, while the sound was being played. The sound could be replayed as many times as needed. Once subjects were satisfied that the goal had been met, selecting the "Next" button took them to the next training example.

[1] The sampling rate of 22.05 kHz. limited the highest sampled frequency to 11 kHz., so the tenth band at 16 kHz. (topmost) was disabled.

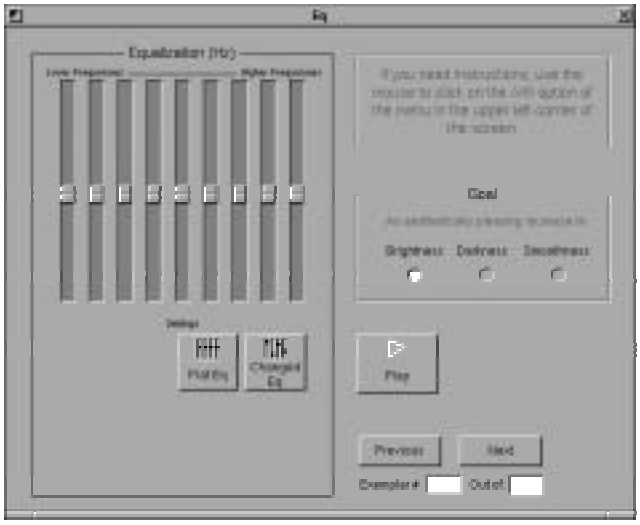

Figure 6: Interface for training the system. Slider changes corresponding to the presented goal are recorded by the system.

For each user, for each of the sound-goal combinations, the computer then created and stored an example consisting of:

1. Soundfile name
2. Goal (one of 3 from the goals window)
3. Final slider positions (scaled from 0 to 31 for each slider)
4. Energy-per-band "signature" for that sound.

These examples were accumulated in the database, embodying the system's knowledge.

4.5 Testing Phase

The 17 subjects from the training phase were evaluated to select the best 11 subjects to continue with the testing phase. These 11 were determined by analyzing the extent to which they moved the sliders. Users who had to move the sliders to an extreme in order to effect a perceptible change in the sound were eliminated. As expected, it turned out the better the subject's hearing as measured by the hearing test, the less the subject tended to move the sliders. The accumulated data for the 11 selected subjects became the dataset of 451 examples per each of the three goals, for a total of 1353 examples, embodying the "knowledge" of the system.

The interface screen for the testing phase is shown in **Figure 7**. Users were asked to give a rating to each of three different equalizations presented by the computer. These three choices were:

1. A linear average
2. The NN average
3. No change

The "No change" equalization was used as a control. Each of these equalizations was represented by one of the Eq Selection Buttons. The linear average was the mean slider change across all 11 users for all 41 sounds for the current goal. This average embodied the approach of "always do the same thing" for a desired goal, such as always increasing the rightmost sliders to make a sound more bright.

At the other extreme the NN average was the mean slider change of the 2 nearest neighbors from that subject's training session only. The nearest neighbors were computed by comparing the example's signature (energy per each of the 9 bands) with the signatures of the stored data. This was essentially placing the example point in a 9-dimensional space and finding the two closest points. The correspondence between the above three types of equalizations and the eq selection button positions were randomized on each presentation.

To start the sound playing, subjects selected one of the Eq selection buttons, "Eq A", "Eq B", or "Eq C," which triggered the sound to start being played. This selected equalization was then compared to the original sound by clicking on the "Flat Eq" button. Each of the equlizations was then given a rating using the slider below it as to how good of a job it did at giving an aesthetically pleasing increase in the highlighted goal. Once the user was satisfied with his or her eq ratings, selecting the "Next" button advanced to the next example and corresponding goal. The eq selection button for "no change" was actually an identical setting to the "Flat Eq" button, a fact that was not always recognized by the subjects.

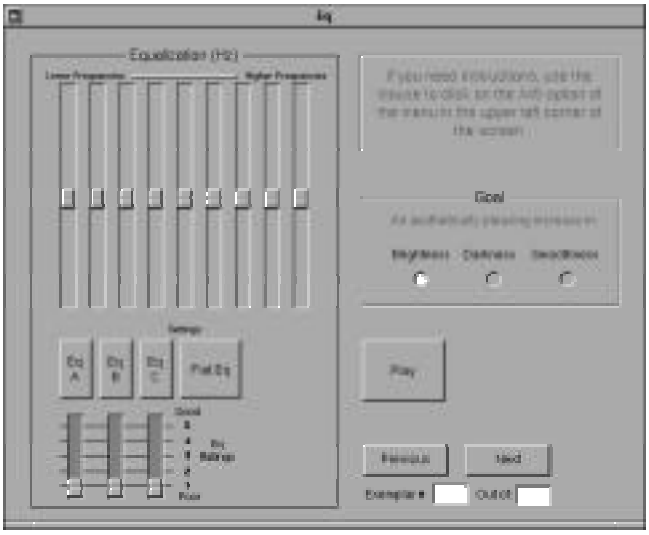

Figure 7: Interface for testing phase. Three possible equalizations are rated as to how well they do in giving an aesthetically pleasing increase in the highlighted goal.

4.6 Results

The experiment validated the hypothesis that nearest neighbor pattern matching (context dependent) does a better job at equalizations than does a linear average. Although the

Eq Ratings sliders are labeled on the screen from 1 to 5, they actually mapped to values from 1 to 15. The mean evaluation of the "no change" equalization (the control) was 2, the linear (non-context dependent) equalization mean rating was 6. The nearest neighbor (context dependent) equalization was over 10.08, which is 68% better than the linear equalization.

Brightness was the easiest timbre for subjects to identify, followed by darkness, with smoothness being the most difficult. Rank ordering of which of the three equalizations was preferred showed than NN equalizations were preferred 81% of the time.

5. CONCLUSIONS

The trained system developed here has been implemented as an expert equalizer (**Figure 8**), where a sound is selected, and then simply by moving a slider under the desired goal, a context-dependent appropriate amount of equalization is done.

One way to look at the system is that it implements a many-to-one mapping, putting many complicated controls into a single control that appropriately affects the outcome. The paradigm presented here could be used to exploit the computer as a tool in extending users' perception in the modalities of sight or smell or other applications in hearing. Using nearest neighbor for a perceptive task could be used by airlines in interpreting video or x-ray data in explosives detection in luggage [7] or by the Navy in interpreting audio signals for submarine detection.

As illustrated in this work's application to sound equalization, a computer can be used as a perceptive tool to give a user an expert level of skill.

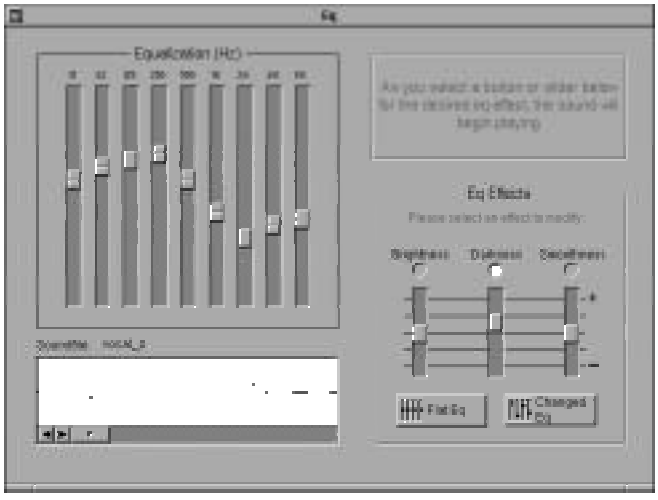

Figure 8: Expert equalizer interface. Slider changes in the "Eq Effects" window for a particular goal automatically give a context-dependent equalization.

6. ACKNOWLEDGMENTS

Thanks to Orion Poplawski, Timothy Mills, and Dave Angulo for assistance in programming. Thanks to Doug Jones for testing speakers and to Peter Langston for providing sound files. Thanks to the following for the many hours of work training and testing the system: Tom Miller, Rob Motsinger, Jeff York, Moses Ling, Helen Hudgens, Shaun Morrison, David Schuman, Mike and Lisa Danforth, Dick Cutler, Jeff Cline, Pablo Perez, Stan Sheft, Norman Kruger, John Bobenko, and John Lanphere.

7. REFERENCES

[1] Bartlett, Bruce, and Bartlett, Jenny. 1995. Engineer's Guide to Studio Jargon. EQ (February): 36-41.

[2] Cohen, Harold. The further exploits of AARON, painter. Stanford Humanities Review 4:2.

[3] Frey, Peter W., and Slate, David J. 1991 Letter Recognition Using Holland-Style Adaptive Classifiers. Machine Learning. The Netherlands: Kluwer Publishers, 6:2 (March).

[4] Holland, John 1986. Escaping Brittleness: The Possibilities of General Purpose Learning Algorithms Applied to Parallel Rule-Based systems. In R.S. Michalski, J.G. Carbonell, & T.M. Mitchell, eds., Machine Learning II. Los Altos, CA: Morgan Kaufman.

[5] **http://www.ncsa.uiuc.edu/Vis/Projects/Docker.**

[6] McCorduck, Pamela. 1991. AARON's code: meta-art, artificial intelligence, and the work of Harold Cohen. New York : W.H. Freeman.

[7] Murphy, Erin E. 1989. A Rising War on Terrorists. IEEE Spectrum, 26:11:33-36.

[8] Quinlan, J. R. 1986. Induction of Decision Trees. Machine Learning 1:81-106.

[9] Rudnicky, Alexander I., Hauptmann, Alexander G., Lee, Kai-Fu. Survey of Current Speech Technology. Communications of the ACM 37:3 (March): 52-57.

[10] Spector, Lee. 1995. International Joint Conference on Artificial Intelligence 95. Montreal, Canada, August 20-25. Workshop on AI & Music. In Press.

[11] Stanfill, C., and Waltz, D. 1986. Toward memory-based reasoning. Communications of the ACM, 29:1213-1228.

[12] Thorpe, C., Herbert, M., Kanade, T. and Shafer, S. 1987. Vision and navigation for the Carnegie-Mellon NAVLAB. In Annual Review of Computer Science. Vol. II. Annual Reviews Inc., Palo Alto, Calif.