New, freely-available technology for studying the impact of simple musical training on speech perception in cochlear implant users

Thomas R. Colgrove,[*1] Aniruddh D. Patel, [#2]

[*] *Departments of Computer Science and Music (BS 2015), Tufts University, USA*
[#] *Department of Psychology, Tufts University, USA*
[1]trcolgrove@gmail.com, [2]a.patel@tufts.edu

ABSTRACT

Cochlear implants (CIs) provide good understanding of speech under ideal conditions (e.g., in quiet), but most CI users have difficulty perceiving pitch patterns accurately. Good pitch pattern perception is essential for the perception of musical melody and speech intonation. Current CI technology does not provide sufficient detail for accurate pitch processing. However, auditory training has been shown to benefit CI users' speech perception, raising the idea that training could also be used to enhance pitch pattern perception. We wish to determine if musical instrument training (learning to play simple pitch patterns on a piano keyboard) enhances the perception of pitch patterns in music and speech. We hypothesize that auditory-motor keyboard training will be more effective at improving pitch pattern perception than purely auditory training, because playing pitch patterns on a musical instrument creates an action-perception link between the motor and auditory systems. To this end, we present a new software platform called 'Contours', which provides simple musical keyboard training in a format aimed at individuals with no prior musical experience. The software runs on an Android tablet connected to a portable 3-octave MIDI keyboard and employs a visual representation of melodic contour shapes that aims to eliminate the steep learning curve of traditional music notation. Additionally, the software records the user's accuracy in playing the contours, provides feedback to the user throughout the training, and saves performance data to a web server for data analysis. The length and difficulty of contours can be gradually increased over the course of training, and several different sounds can be used, from simple sine waves to piano tones. By testing auditory perception before and after training, one can see if training using Contours enhances the perception of pitch patterns, and of music and speech more generally.

I. INTRODUCTION

Although cochlear implants provide good understanding of spoken words under ideal conditions (e.g., speech in quiet, familiar voices), most cochlear implant (CI) users have great difficulty perceiving pitch patterns accurately (Limb and Roy 2014). Good pitch pattern perception is essential to music appreciation and to understanding the "melody of speech" (the ups and downs of voice pitch, which help signal phrase boundaries and emotion). Current CI technology does not provide sufficient detail for accurate pitch processing. Much research and development has been directed at improving the spectro-temporal resolution of the CI (e.g., "virtual channels" between the implanted electrodes, high-rate stimulation), but with little-to-no improvement over standard implant technology. However, auditory training has been shown to benefit CI users' speech perception. This raises the idea that training could also be used to enhance pitch pattern perception.

In fact, research has shown that purely perceptual training can improve pitch pattern perception in CI users (Galvin et al., 2009). This training involves matching simple pitch contours to their visual analogs, based on Galvin et al.'s (2007) "Melodic Contour Identification" (MCI) test. In this test, a listener hears one of 9 pitch contour shapes (e.g., rising, falling, rising-falling) and identifies it by clicking on the corresponding image on a computer screen (see Figure 1 below). This is an easy task for normal listeners but a very difficult task for CI users, due to their problems with pitch pattern perception. Galvin et al. (2009) showed that by training on the MCI task (using different root notes than used for testing), CI users can improve their performance on the MCI test.

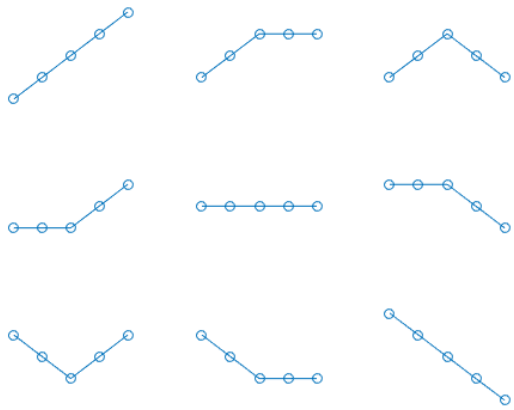

Figure 1. Schematic of the nine pitch contours in the Melodic Contour Identification (MCI) test (Galvin et al., 2007). After hearing a pitch pattern, the listener identifies the pattern by choosing the corresponding contour on a computer screen (e.g., rising, falling, rising-falling etc.). Each pattern has 5 pitches, and the pitch distance between notes in a contour can be systematically varied from small to large. These serve as the easy contours in the Contours program.

The current research is motivated by the idea that CI users can enhance their pitch contour perception by training that involves **playing** pitch contours on a piano keyboard. We hypothesize that keyboard training will prove more effective at improving pitch pattern perception than purely auditory training. This is because playing pitch patterns on a musical instrument (vs. simply hearing them) creates an action-perception link between the motor and auditory systems of the brain, which is absent in purely perceptual training (Lahav et al. 2007; Chen et al., 2012). There is evidence from auditory neuroscience that training involving action-perception coupling leads to more effective neural changes in sound processing than does training based on listening alone (e.g., Lappé et al., 2008; Matthias et al., 2015). This empirical work provides the background and rationale for our hypothesis.

The feasibility of this type of training has been demonstrated in a pilot study in which two CI users

underwent this type of training (see Patel, 2014 for details). Training consisted of playing patterns based on the pitch contours in Figure 1, using either 5 successive white keys (1-2 semitone spacing) or black keys (2-3 semitone spacing) on a piano keyboard. These spacings were deemed optimal as many CI users have difficulty perceiving pitch differences less than 2 semitones. Thus the music training aimed to develop greater precision in pitch contour processing than the CI users normally have, using a form of sensorimotor training in which the participants themselves produce the auditory pattern they are learning. After keyboard training, both participants improved on the MCI test. One participant also showed an improvement in speech-in-noise perception, while the other showed an improvement in statement-question identification based on linguistic pitch contour (data reported in Patel, 2014).

To facilitate further research of this type, we have developed a new software program ('Contours') which interfaces with a MIDI keyboard and trains a user to play melodic contours of different shapes, while tracking progress by logging performance data. No prior musical experience is assumed of the user. This software is considerably more flexible than the software used in the study of the two CI users reported in Patel (2014). For example, the new software offers three different levels of contour complexity, allowing the user to progress from simpler to more challenging contours, which should help keep the training interesting and motivate participants to practice. Easy contours are the contour shapes in Figure 1: these have 5 notes and at most one direction change in the contour (e.g., from rising to falling, or rising to flat). Medium difficulty contours are shown in Figure 2: these have 6 notes and 2 contour direction changes. Hard difficulty contours are shown in figure 3: these have 7 notes and 3 contour direction changes. At each level of difficulty, the program presents each contour shape at several different transposition levels, allowing the user to play the contour across a range of frequencies.

Another aspect of the flexibility of Contours is that at each difficulty level, the user has options for the size of pitch gaps between notes in the contours. For CI users, different pitch contours shapes are considerably more challenging to discern when the gaps between notes are small vs. large (see Galvin et al., 2009, Figure 3). In Contours, pitch gaps can be set so that the gap size between pitches is 0, 1, or 2 white notes on the keyboard. (To simplify the playing of the piano keyboard for non-musicians, black keys are not used in during the training.)

One final way in which Contours is flexible is that the user is given 5 choices for the sound produced by the keyboard: sine-waves (the easiest sounds for CI users to perceive), square waves, triangle waves, sawtooth waves, and sampled piano tones. Thus users can learn what a pitch contour sounds like when played in different timbres, which will hopefully allow generalization and recognition of contours when created by different sound sources (including the human voice).

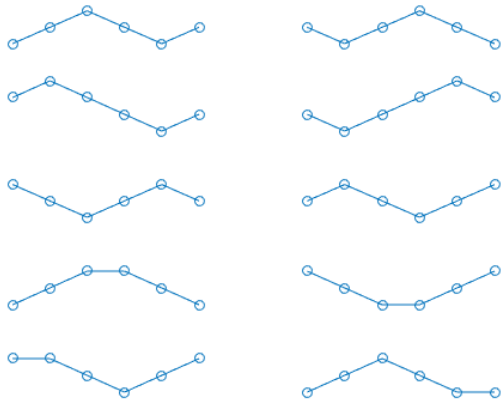

Figure 2. Schematic of the 10 pitch contours in the medium difficulty condition of the Contours program. Each pattern has 6 pitches and 2 changes of contour direction (e.g., rising to falling, level to falling). The pitch distance between notes in a contour can be systematically varied from small to large.

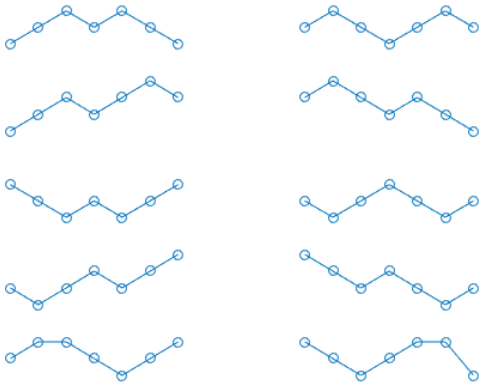

Figure 3. Schematic of the 10 pitch contours in the hard difficulty condition of the Contours program. Each pattern has 7 pitches and 3 changes of contour direction (e.g., rising to falling, level to falling). The pitch distance between notes in a contour can be systematically varied from small to large.

Apart from the flexibility of Contours, another other important feature of the program is data logging, which records the timing and accuracy with which each contour is played, allowing researchers to quantify the improvement in contour performance over time (for details, see the next section: unit testing revealed that logged times have an accuracy of +/- 100 ms). Importantly, Contours has a game-like visual interface, which presents contours as visual shapes using attractive graphics, dynamically shows the user which note to play next in a contour, and gives visual feedback each time a mistake is made or a contour is played correctly.

Contours is freely available and open-source, allowing researchers to customize it to their needs, e.g., by adding other contours shapes, sounds, or gaps sizes between notes.

II. DETAILS OF TRAINING

A.Notation

An important feature of Contours is the simplicity and readability of the musical notation. While traditional music notation has a steep learning curve, the notation in Contours uses color as a visual aid to assist users in playing the patterns presented (Figure 4), and does not require learning note names (e.g., C, D, E, etc.) The notation uses unstemmed circles to represent pitches. Each circle has a specific color, which corresponds to a color on the MIDI keyboard (Figure 5). The notation and keyboard span a 3 octave range (from C2 to C5, i.e., MIDI note 48 to 84), with the pattern of colors repeating each octave (e.g., all Cs on the keyboard are purple, across different octaves).

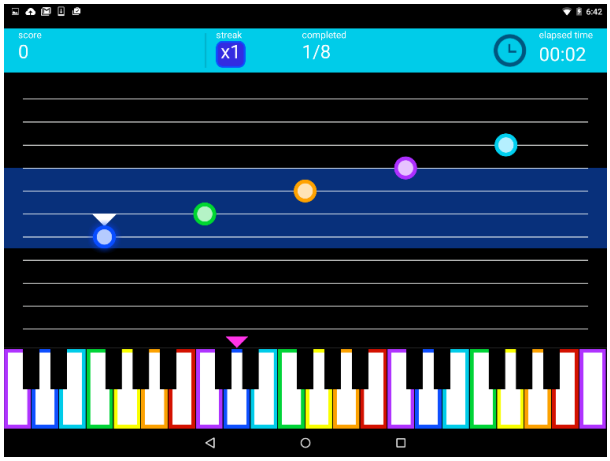

Figure 4. Screenshot of Contours, presenting an easy contour to the user. The next note to be played is indicated by an inverted wedge over the circle in the music notation, and over the key that should be pressed on the keyboard to produce that note. When the corresponding key is pressed, the wedge moves to the next pitch in the music notation and the corresponding key on the image of the keyboard. The image of the keyboard at the bottom of the Contours screen is for visual reference only: the user plays a physical keyboard attached to the tablet, which has the same color scheme (see Figure 5).

Figure 5. Contours running on an Android Tablet, connected to the color-coded 3-octave MIDI keyboard. Colors are placed on the white keys using adhesive paper (no black keys are used in training). Each note within an octave has a unique color, and the color pattern repeats across octaves.

As with Western music notation, the staff lines and spaces map in a one-to-one fashion to the large keys of the keyboard (i.e., the white keys, whose tips are covered with colors, cf. Figure 5). Thus the contour shown in Figure 4 involves playing an ascending contour which starts on a large blue key (near the middle of the keyboard) and skips one large key between each of its pitches. An important difference between the music notation used in Contours and Western music notation is that the notation in Contours does not include any accidentals. (Correspondingly, none of the black keys are used during training.) This is to maintain the simplicity of notation and make it easy to learn for individuals with no prior musical training.

B. Training Program Flow

i. Each time a user plays Contours, they select a contour difficulty (easy, medium or hard, cf. Figs 1-3), a pitch gap size between successive notes in the contours (narrow, medium, or wide), and a sound for the application to play (sine wave, triangle wave, square wave, sawtooth wave, or sampled piano sounds). For pitch gap size, narrow, medium and wide correspond to the number of colored keys skipped when playing consecutive notes in a contour: i.e., 0, 1, or 2, respectively. (Thus the ascending contour shown in Figure 4 has a medium gap size).
   i. There is no enforced order in which the user progresses through the different conditions. This allows the researcher suggest a particular order, e.g., easy level first, then medium, then hard, with wide pitch gaps at all levels, followed by repeating these three levels with medium pitch gaps, and then again with narrow pitch gaps, all using sine-wave tones. This example would result in 9 conditions (3 contour difficulty levels x 3 pitch gap sizes), all using the sine-wave sound. Repeating this regimen with another sound (such as the square wave) would add 9 more conditions.

ii. Once the user chooses a difficulty level, a pitch gap size, and a sound, Contours presents the user with a block of contours consisting of each contour shape from that difficulty level at 5 different transpositions. (Hence a block has 45 contours at the easy level, and 50 at the medium and hard levels, cf. Figs 1-3.) For a given contour shape, transpositions are chosen randomly (without replacement) from all possible transpositions that fit within the 3 octave range. Thus, 5 different transpositions are always presented, and repeating a block will result in a new set of randomly-chosen transpositions.

iii. When playing a given contour, if the user makes a mistake (e.g., plays a wrong note), a visual message appears in the center of the screen (e.g., “Whoops!”) and the user must begin the contour again from the first note. Upon playing a contour correctly, the user gets a visual congratulation message (e.g., “Good job!”) and earns points, which appear in the top bar of the screen. (See

section on Data metrics for a description of how points are calculated.)

iv. Once a block of contours is completed, the user is asked a few short questions about the contours played, and responds to each question using a number from 1 to 5. The questions are:
   i. How difficult was the task?
   ii. How different did the contours sound from each other?
   iii. Do you think you are improving at the task?

v. Once a block is completed, quantitative data about the user's performance on each of the 45 (or 50) contours is logged, such as number of errors, speed and accuracy. These data are stored locally on the Android device, and sent to a remote server in JSON format (see section on Data metrics for details). These data can be downloaded from the server as spreadsheets in .csv format for analysis. This allows researchers to quantify the progress of users over multiple sessions. (Responses to the questions described above are also stored.)

C. Data Metrics

Within a block, the user earns points for each correctly-played contour. Points reflect both speed and accuracy of performance. The score for a contour = 100 points (for completing the contour correctly), plus a bonus which equals 100 minus the total time it took to complete the contour (in milliseconds/250). This number is then multiplied by an integer which corresponds to how many contours have been completed in a row without any errors (the maximum multiplier is 8). Finally, this number is added to the existing score to create a running point total for the block. After completing a contour, the top bar briefly displays the points earned for that contour, and the running total for the block, with the latter score remaining on screen throughout the block.

When a block is completed, performance data about that block are stored locally on the tablet, and then uploaded to a server when an internet connection is available. These data can then be downloaded from the server as a .csv spreadsheet by the researcher for analysis, e.g., to quantify improvement over time. Each spreadsheet has 11 columns. The columns are as follows:

1. Contour ID: an integer between 1 and 9 (for the easy contour condition) or 1 and 10 (for the medium and hard conditions), specifying contour shape. For the easy condition, the numbers correspond to the layout of shapes in Figure 1, in the following order:
1 2 3
4 5 6
7 8 9
For the medium and hard conditions, the numbers correspond to the layout of shapes in Figure 2 and 3, respectively, in the following order:
1 2
3 4
5 6
7 8
9 10

2. Difficulty: Easy, medium or hard (cf. Figs 1 – 3)

3.Note gap: an integer between 0 and 2, indicating how many colored keys are skipped between consecutive notes of the contour.

4.**Sound**: specifies which of the 5 sounds was used: sine wave, square wave, triangle wave, sawtooth wave, sampled piano.

5. Start note: The MIDI note number of the first note of the contour (indicates the transposition level).

6. Total completion time: Time from when the contour is first posted on the screen to when it is completed correctly. If mistakes were made, and the user had to start the contour again, this time is included. (However, time taken by the posting of messages triggered by errors, e.g., "Whoops!", is not included.)

7. Num errors: The number of wrong notes played before completing the contour correctly.

8. Percent error: Num errors/number of notes played while completing a contour.

9. Completion time (correct run): Time between the onset of the first note and the onset of the last note when the contour is played correctly

10. Note interonset interval sd: The standard deviation of durations between note onsets when the contour is played correctly.

11. Date and time
Date and time the contour was completed.

Details on how to access data spreadsheets from a server are provided on the GitHub repository for the Contours project.

III. TECHNICAL SPECIFICS AND EXTENSIBLITY

The Contours training application for Android can be downloaded from GitHub and installed on any Android tablet device, although tablets with at least 9" of screen space, running Android 5.0 or higher are recommended. This tablet should be connected to a 3-octave MIDI keyboard to which colored adhesive paper has been added to match the color scheme in the software (see Figure 5). We used a CME Xkey 37.

The application is open source under the MIT license (https://opensource.org/licenses/MIT), and can be freely used, shared, and modified to suit different testing needs. While there are many possible extensions of the program, there are a few that can be made with relatively minimal coding (see below). We chose Android as the operating system for its portability and affordability, the open nature of the platform, and the ability to use the application on many different devices.

A. Modification of Contour Shapes in XML

The preloaded musical contour shapes are defined in a project xml contours.xml file, that can be easily modified to provide other contours of different lengths and shapes. This allows researchers to easily expand on or replace the preloaded set of contours. For example, some users may want to progress beyond the preloaded contours to more challenging and complex shapes. Each contour shape only needs to be specified once, as the software will automatically transpose the contour to multiple different pitch levels. Example xml codings of three contours are shown below. Note that contours are specified using standard note names from Western music

(e.g., C2, E2), which combine a pitch class and an octave number. (Further documentation on code specifics can be found on the project github.)

```
<!--Rising one skip -->
    <item>C2,E2,G2,B2,D3</item>
<!-- Rising Flat one skip -->
    <item>D2,F2,A2,A2,A2</item>
<!-- Rising Falling one skip -->
    <item>C2,E2,G2,E2,C2</item>
```

3 *xml formatted contours, 5 notes each*

B. Modify Application Timbre using Pure Data

One powerful component of Contours is the ability to test the effect of various timbral qualities on pitch perception and pitch contour learning among CI users. The synthesizer component of the application uses the Pure Data visual language to enable the generation of a wide array of different musical sounds. The integration of Pure Data with Contours is modular and allows modification and swapping of pure data patches. It is straightforward to modify or swap out the currently included Pure Data *patches*, to use an expanded set of sounds. The two currently included modules are a fully functional subtractive synth based on a related implementation written by Christopher Penny, and an instrument sampler. Possible modifications include modifying the existing subtractive synth, changing the sample sets available on the sampler, or to using a different patch altogether. More detailed instructions for extending the Contours synthesizer can be found in the project documentation.

C. Example Server

The Contours application comes preloaded with the capability to send detailed test results to a server application in JSON format. A fully-featured example server that can be used with the contours server is provided on the application GitHub. The server is coded in a Flask/Python/MongoDB back end. The example server also includes a simple front end webpage which displays summary test results in tabular form, and enables the spreadsheet download of results for each completed block of contours in CSV format. Administrators of the Contours training program may use the example server as provided, extend it, or use their own server.

D. Further Extensions

More extensive changes to the contours application require some knowledge of Java and the Android SDK. Brief descriptions of possible future extensions will be detailed in the following section.

IV. PROPOSALS FOR FUTURE CAPABILITIES

The Contours application currently tests the effects of simple musical keyboard training on CI users ability to perceive pitch contours. However, there are multiple ways in which this application could be significantly extended. For instance, the rhythm and cadence of speech is arguably as important as pitch variation. Furthermore, rhythm is often a fundamental component of people's ability to identify different melodies. It is possible that introducing a rhythmic component to the training program could be of greater benefit to CI users in understanding speech.

Furthermore the keyboard training at present only presents the user with monophonic melodic patterns. It would be fairly straightforward to extend the training to include contours featuring chords, or more than one note played at the same time.

V. CONCLUSION

Contours provides a flexible, sophisticated platform for training CI users to play simple melodic contours, with the goal of improving their pitch contour perception, with consequent benefits for both music and speech perception. The software assumes no prior musical knowledge or experience. Existing perceptual tests, such as the melodic contour identification (MCI) test (Galvin et al., 2007), can be given before and after training to determine if learning to play melodic contours enhances the ability to perceive and discriminate pitch contours. Ultimately, one could compare such training to purely auditory training (matched in duration) to test the hypothesis that actively playing contours is more effective for enhancing pitch perception, due to the auditory-motor nature of such training.

The software is available for download at https://github.com/contoursapp/Contours

ACKNOWLEDGMENT

This research was supported by a grant from the Paul S. Veneklasen Research Foundation. We thank Allison Reid for her help in developing the medium and hard difficulty contours and test the Contours software, and Paul Lehrman and Chris Penny for their technical insights.

REFERENCES

Galvin, J. J., Fu, Q. J., & Nogaki, G. (2007). Melodic contour identification by cochlear implant listeners. *Ear and hearing*, *28*, 302-319.

Galvin, J. J., Fu, Q. J., & Shannon, R. V. (2009). Melodic contour identification and music perception by cochlear implant users. *Annals of the New York Academy of Sciences*, *1169*, 518-533.

Lappé, C., Herholz, S. C., Trainor, L. J., & Pantev, C. (2008). Cortical plasticity induced by short-term unimodal and multimodal musical training. *The Journal of Neuroscience*, *28*, 9632-9639.

Lahav, A., Saltzman, E., & Schlaug, G. (2007). Action representation of sound: audiomotor recognition network while listening to newly acquired actions. *The Journal of Neuroscience*, *27*, 308-314.

Limb C.J., & Roy A.T. (2014). Technological, biological, and acoustical constraints to music perception in cochlear implant users. *Hearing Research*, *308*, 13-26.

Mathias, B., Palmer, C., Perrin, F., & Tillmann, B. (2015). Sensorimotor learning enhances expectations during auditory perception. *Cerebral Cortex*, *25*, 2238-2254.

Patel, A. D. (2014). Can nonlinguistic musical training change the way the brain processes speech? The expanded OPERA hypothesis. *Hearing research*, *308*, 98-108.