An Adaptive Gesture-to-Speech Interface for DIVA

Allison Lenters
MAGIC Lab, UBC

1. Introduction

The goal of this project is to create an adaptive interface for the DIVA project, that uses statistical functions to perform gesture-to-speech mapping. This will replace the current static gesture-phoneme mapping, improving ease of use of the system, and increasing the fluidity and intelligibility of the speech produced. The interface is realized as a pair of Radial Basis Function Networks (RBFNs), which produce vowel and consonant sounds respectively, and which are interpolated by a third Vowel/Consonant decision network, to produce a single continuous stream of speech sounds.

2. Description of the system to be implemented

DIVA is a system that performs continuous hand gesture-to-speech mapping, using a number of hardware items to track a performer's hand movement and gestures, and a parallel-formant speech synthesizer to output the speech sounds. Performers train for roughly 100 hours to learn the gestures [1]. Differing from sign language, in this system, speech is produced phonemically: each gesture corresponds to a single phoneme. The gesture-phoneme mapping was determined previously [1], and corresponds roughly to the layout of the human vocal tract and articulators. The system can be considered to be an *artificial vocal tract* (AVT). Consonants are produced by touching fingers to the thumb, much like how consonant sounds are produced in the mouth by touching the tongue to articulators. Vowels are produced by moving the hand around in horizontal space, in a layout that is very similar to the vowel space of the mouth.

A performer's gestures are tracked by a number of hardware components: A CyberGlove, worn on the hand performing the gestures (the right hand), measures 18 angles of the performer's hand. The Polhemus Fastrack, worn on the right arm, measures 6 degrees of freedom of the performer's hand (X, Y, and Z position; yaw, pitch, and roll). Each of the outputs from the CyberGlove and Polhemus Fastrack are scaled to be between 0 and 1. A contact glove, essentially a modified keyboard, is worn on the performer's left hand. It is used to produce the

stops (B, D, G, J, P, T, K, CH), which proved to be too difficult to perform using gestures due to their rapid release rates. Finally, the performer uses a foot petal to control the master volume of the sound produced by the system.

Although the learning curve is steep, the system is intended to be used by a multitude of performers, each with her own unique hand shape and performance style. It is expected that as each performer interacts with and gets comfortable with the system, she will modify and improve her performance style. As such, an adaptive interface is used for the gesture-to-formant mapping. Each performer can create her own "accent," by providing multiple training samples for each phoneme. The system trains on these samples, and creates a set of accent files, which can then be used by the system in performance mode, to execute the gesture-to-phoneme mapping. As time goes by and the performer improves, she can re-train the system, modifying her accent to better suit her performance style.

DIVA's synthesizer requires 10 *control parameters* to generate a sound: nasal amplitude (ALF), the frequency and amplitude for the first, second, and third formants (F1, A1, F2, A2, F3, A3), high-frequency amplitude (AHF), degree of voicing (V), and fundamental Frequency (pitch; F0). These control parameters must be produced by the performance module and sent to the synthesizer every 10 ms.

In performance mode, the system performs gesture-to-speech mapping. There are 26 possible phonemes (not including the 8 stops) and one gesture corresponding to each one. Each phoneme also has a set of corresponding control parameters, which were determined through careful study [1]. If, for example, the "E" control parameters were sent to the synthesizer, the system would play back a pure "E" sound. However, the goal of the system is not to segment the gesture stream into discrete gestures, classify each one, and output its corresponding phoneme. This could be described as continuous-to-discrete mapping, and it would produce an extremely disjointed speech signal. Rather, the system performs continuous-to-continuous mapping, in which, at any given instant, the gesture currently sensed by the hardware is mapped to a *combination* of discrete phonemes, depending on how similar the gesture is to each of the phonemes' target gestures. Because the frequency and amplitude scales of the control parameters are linear, control parameters can be easily combined to produce new sounds. For

example, simply taking the arithmetic mean of control parameters for two phonemes will produce a sound that sounds "in between" the two phonemes.

While the frequency and amplitude scales themselves are linear, the function used to determine the weights for the weighted sum (discussed later) is not. As a result, the change in control parameters produced in moving from one gesture to another is not linear, but rather sigmoidal in nature. This is acceptable because, although the frequency scale itself is linear, the human ear does not hear it as such. When listening to a tone that shifts from a low frequency to a high one, the human ear will segment the stream into a number of discrete notes. The same applies for shifting from one phoneme to another. The ear will hear a clean break between one phoneme and the next. The method used by the system for combining phonemes is called *normalized non-linear interpolation*.

To produce the control parameters, the system utilizes three different networks: the Vowel network, the Consonant network, and the Vowel/Consonant (V/C) Decision network. The Vowel network is a 2-11-8 feed-forward network that produces vowel sounds based on the glove's X-Y position. The 2 inputs are the scaled X and Y positions. The 11 hidden units are the 11 vowel phonemes (EE, I, E, AA, U, AR, O, AW, OO, UU, A). The outputs are the 8 variable control parameters for vowels (ALF, F1, A1, F2, A2, F3, A3, AHF). The Consonant network is a 10-15-9 feed-forward network that produces consonant sounds based on the gestures performed by the right hand. The inputs are 10 scaled values from the CyberGlove (2 flex angles for each of the thumb, index, middle and ring fingers; thumb abduction and rotation angles). The 15 hidden units are the 15 consonant phonemes (W, L, R, V, F, DH, TH, Z, S, ZH, SH, H, M, N, NG). The outputs are the 9 variable control parameters for consonants (ALF, F1, A1, F2, A2, F3, A3, AHF, V). The V/C Decision network combines the control parameters produced by the other two networks. If the hand is completely open, a pure vowel sound is produced. If the hand is closed (i.e., some of the fingers are touching), a pure consonant sound is produced. Otherwise, the control parameters from the Vowel and Consonant networks are interpolated based on the degree of openness of the hand. The design specifics for the V/C Decision network in DIVA are yet to be determined and will not be discussed here.

Unlike previous versions of the system, all gesture-phoneme mappings will be learned from training samples, including the positions of the vowels in X-Y space. The system will adapt to

each performer's unique performance style, rather than forcing the performers to adapt their performance styles to the system. The two main benefits of this system will be an unlimited vocabulary, and much greater control over intonation than is possible using traditional speech synthesis systems.

A second divergence from previous versions of the system is that the bounds used for scaling each input parameter are no longer be set explicitly. In the past, these were set by the performer. In the case of the vowel space, she would set bounds on the four edges of the horizontal plane. In the case of the consonant space, she would wiggle the fingers of her right hand, and the system would record the maximum and minimum bend angles for each finger. This produced unexpected behaviour if the bound values were not actually the correct bounds for the data in the accent file. In DIVA, the bounds are calculated dynamically, as part of the training function; they correspond to the maximum and minimum values of all of the training data considered for each dimension. Therefore they are guaranteed to always be correct.

3. History of the project

The Glove-TalkII was developed in the early 1990s by Sid Fels. Its predecessor, Glove-Talk, had used the continuous-to-discrete mapping described above. It was quickly replaced by Glove-TalkII, which performed continuous-to-continuous mapping using an adaptive interface similar to the one that the topic of this paper (albeit Glove-TalkII utilized a simple static mapping of the vowel space).

Glove-TalkII was later refactored by Bob Pritchard of the GRASSP Project, into Max/MSP code. The intentions of GRASSP were to make the system easier to use for non-programmers, to increase the system's capabilities in terms of audio output, and to introduce the possibility of video output [2].

One aspect of Glove-TalkII that was lost in the transition to GRASSP was the adaptive interface. In GRASSP, performers create their "accent" by providing a single training sample for each gesture. The result of this simplified training paradigm is that, although each gesture only involves a few fingers, the performer has to remember and reproduce the positions of all five fingers for each gesture. This has a number of disadvantages compared to an adaptive interface. Firstly, it places a greater burden on the performer's memory, slowing down the learning process

and increasing the frequency of errors. Additionally, requiring the performer to move all of her fingers into position for each gesture greatly reduces fluidity and decreases sound quality. Over various instances of a single gesture, the fingers not involved in that gesture may be coming from a number of different positions, depending on what the previous gesture was. In consequence, it becomes extremely difficult for the performer to produce exactly the intended sound, and the intelligibility of the speech produced decreases a considerable amount.

Under the DIVA grant, sponsored by NSERC and Canada Council for the Arts, GRASSP will be refined and expanded, preparing it for use in onstage performances and as the subject of research projects. One imperative component of the improvements is the re-introduction of the adaptive interface, and it was for this purpose that the current project was undertaken.

4. Methods

Describing the problem in mathematical terms, the gesture-to-sound map is in 2- or 10-dimensional space (for vowels and consonants). However, especially in the case of the consonants, only a subset of those dimensions are required to take on specific values for each phoneme. The values of the dimensions not involved in a specific gesture should not affect the sound produced. Gaussian Radial Basis Function Networks (RBFNs) are well-suited to solving this problem. RBFNs exist in N+1-dimensional space – one dimension for each of the N inputs to the RBFN, and one dimension for the output of the RBFN. A RBFN contains K hidden units, each of which is defined by two N-sized vectors, one of arithmetic means, and one of standard deviations, for each of the N input dimensions.

In terms of the Vowel network, which is easiest to visualize, there are 11 hidden units – one for each phoneme. The hidden units take two inputs, the current X and Y positions of the glove, and each unit outputs a value, indicating how close the input values are to its center. The Vowel RBFN can be visualized as in Figure 1.

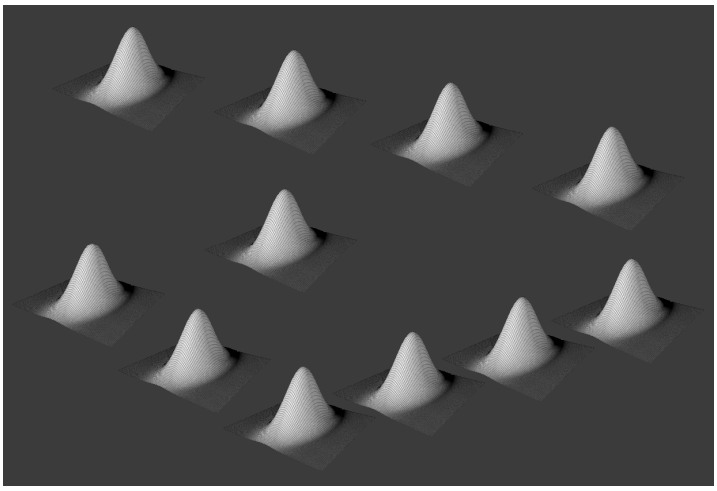

Figure 1: A rough visualization of the Vowel RBFN

The exact positions of the hidden units' centers are determined by the values of the arithmetic means vector. The output function used by the units is a Gaussian function. Thus, the width of the unit along each dimension is determined by the standard deviations vector. In terms of the Vowel network, if the standard deviations for both the X and Y dimensions are very small, the width of the hidden unit in both dimensions will be very narrow, and the glove will have to be moved quite close to a phoneme's center before a favourable value is returned by the corresponding hidden unit in the RBFN. If the X standard deviation is low, but the Y standard deviation is high, the glove will have to hit a very specific location in the X direction, but will have a much greater range of favourable Y values.

The output function used by the hidden units is as follows:

$$\text{hidden\_output}\,(\mathbf{C},\mathbf{R},\mathbf{I}) = e^{-\sum_{i=1}^{N}\frac{(C_i - I_i)^2}{R_i}}$$

Next, the outputs of each RBFN hidden unit are normalized, so that they sum to 1. As a result, the normalized output by each hidden unit comes to represent the probability that the performer is trying to produce that hidden unit's corresponding phoneme, with the closest units returning much higher values than the units that are farther away. For example, if the glove is very close to the X-Y location of the "E" unit, then the "E" unit will return a value very close to 1. If the

glove is halfway between the "E" and "EE" units, each of these will return a value close to 0.5. An added benefit of normalization is that it allows the performer to "overshoot" units on the boundaries of the performance plane, without the sound suddenly dropping off. As the values returned by the hidden units are required to sum to 1, the unit that was overshot will still be closest to the glove, will still return a comparatively high output value, which will become 1 after normalization.

Finally, the RBFN must return a single vector of outputs to the rest of the system. Each phoneme has a set of control parameters, as discussed earlier. The control parameters for each hidden unit's phoneme are combined using a weighted sum, with the normalized outputs of the units acting as weights. Returning to the earlier example, if the "E" unit returns a value of 1 and the rest of the units return 0, the weighted sum will be equal to "E"'s control parameters. If the "E" and "EE" units each return 0.5, the weighted sum will be the arithmetic mean of the control parameters for "E" and "EE."

The full equation used by the performance module to calculate the control values is thus:

$$\mathrm{ft}\,(\mathbf{fs}, \mathbf{C}, \mathbf{R}, \mathbf{I}) = \left\{ \; k=1\ldots K \,\middle|\, e^{-\sum_{i=1}^{N} \frac{(C_{ki}-I_{ki})^2}{R_{ki}}} \right\} \cdot \mathbf{fs}$$

where K is the number of hidden units,
N is the number of input parameters,
C is the vector of arithmetic means for each dimension,
R is the vector of standard deviations for each dimension,
I is the vector of inputs for each dimension,
$\{ \cdot \mid \cdot \}$ creates a 1-by-K matrix of normalized hidden unit outputs,
fs is the K-by-10 matrix of control parameters for each phoneme,
the multiplication performed is matrix multiplication,
and the result is a 1-by-10 matrix of control parameters

This function can be visualized as in Figure 2.

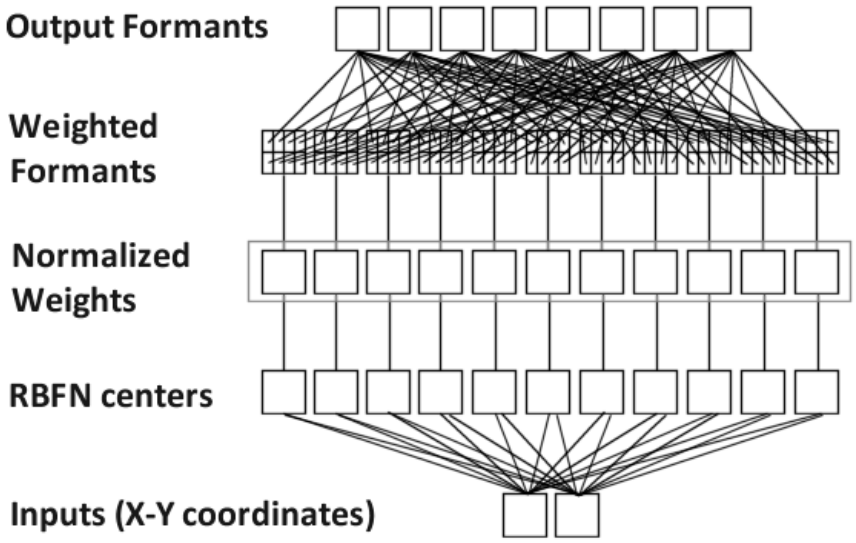

Figure 2: A visualization of the calculations performed by the Vowel RBFN

We have elected to use MATLAB to implement this solution. MATLAB is ideal because it is well-suited to matrix manipulation, which is used heavily in calculations involving RBFNs. This greatly simplifies the code, making it cleaner, and easier to understand and maintain. Of course, MATLAB also increased the steepness of the learning curve – even figuring out how to install it proved to be a challenge. MATLAB can be compiled as C code, which can be wrapped by a MAX object for integration into the rest of the DIVA framework.

5. Results and Discussion

Sid Fels is also working on the DIVA project and had a good idea of what needed to be done to create the adaptive interface for DIVA. One of the main challenges of this project was for me to figure out exactly how he envisioned the solution. The second challenge was to figure out how to implement this in MATLAB, which I had not used prior to the start of this project.

To date, the Vowel network has been completed. The Vowel RBFN can be trained with an Excel "accent" file, and thereafter, when given an X-Y pair, the performance portion of the code will return a set of control parameters. The code has yet to be integrated into the DIVA system, but two preliminary analyses have been performed, as discussed below. The Consonant RBFN

has not yet been completed, but it is expected to be very similar to the Vowel RBFN, with a far greater number of dimensions.

5.1 Numerical analysis

As expected, when the performance function is given an X-Y pair that corresponds exactly to the center of one of the RBFN hidden units, the control parameters returned are exactly equal to those of the corresponding phoneme. Otherwise, the interpolation between hidden units' outputs has been verified to be sigmoidal in nature, as expected.

5.2 Graphical analysis

For each of the 10 control parameters, the MATLAB graphing functions were used to produce 3D graphs of the output values for each pair of X-Y values. The result is a topographic landscape of the output values for each of the control parameters. As can be seen Figure 3, this landscape looks exactly as we might expect it to. Different regions of the horizontal plane correspond to different vowel spaces, and the boundary between these spaces is sigmoidal, although this is more apparent at higher resolutions. It must be noted that the topology of the graph depends greatly on the training data used. The training data being used currently is a modified version of an old accent file, which had only one training sample for each phoneme. It was expanded to include multiple training samples for each phoneme, but all of the new training samples were fabricated. Analysis using real training data will be performed later on in the project, once the exact format of the new accent files has been determined and a new user interface for training has been built. Therefore, this graph is given as proof of concept, but it is essentially meaningless on its own.

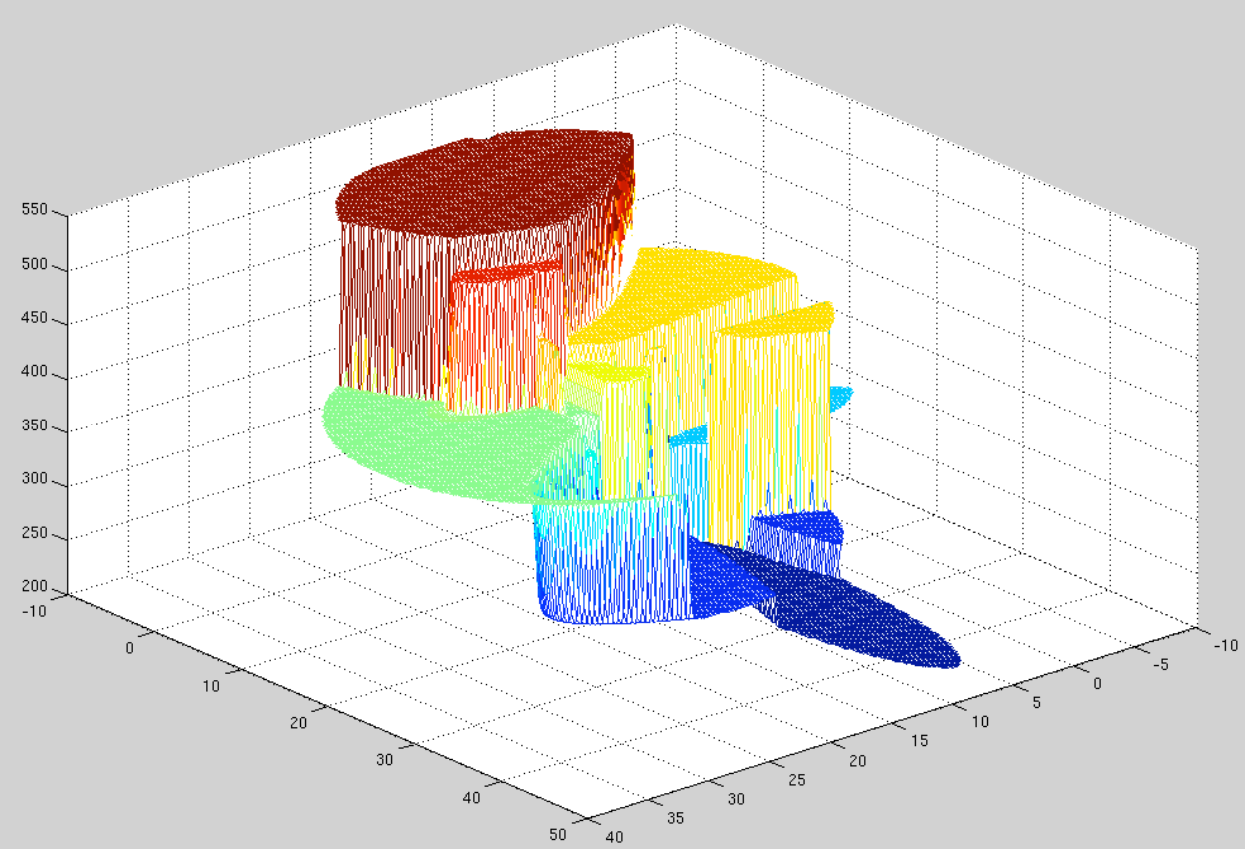

Figure 3: Topographic map of the F1 output from the Vowel network value, against X and Y position, for fabricated training data.

6. Future work

I will continue to work on this project through the summer. The team's goal is to produce a first version of the DIVA system, with all features working, by the end of May. There are many tasks left to complete. First, we must decide on a format for the "accent" files, which are used to pass the trained RBFN from the training module to the performance module. Next, the MATLAB code for the Vowel network must be compiled as C code and tested in the larger context of the complete DIVA system. Additionally, the code for generating the Consonant RBFN must be generated. This should be very similar to the Vowel RBFN and therefore will hopefully be relatively simple to implement. The V/C decision network must be designed and implemented. Finally, a user interface must be written in MAX/MSP that allows users to create training "sessions," select which sessions to train on, and train and retrain their accents.

REFERENCES

[1] Fels, S.S. ***Neural Networks for Computer-Human Interfaces: Glove-TalkII.*** Proceedings of the International Conference on Neural Information Processing (ICONIP96). Pages 1299-1304. Hong Kong. Sept. 1996.

[2] Pritchard, B., &, Fels, S.S. **The GRASSP Environment: Gesturally Realized Audio Speech and Song Performance**. Presented at New Interfaces for Musical Expression 2006, at NIME'06, Paris, France 2006.