The Vocal Joystick: A Voice-Based Human-Computer Interface for Individuals with Motor Impairments[*]

Jeff A. Bilmes[†]***, Xiao Li***[†]***, Jonathan Malkin***[†]***, Kelley Kilanski***[‡]***, Richard Wright***[‡]***,***
Katrin Kirchhoff[†]***, Amarnag Subramanya***[†]***, Susumu Harada***[§]***, James A. Landay***[§]***, Patricia Dowden***[¶]***, Howard Chizeck***[†]

[†]Dept. of Electrical Engineering **[‡]Dept. of Linguistics**
[§]Dept. of Computer Science & Eng. **[¶]Dept. of Speech & Hearing Science**
University of Washington
Seattle, WA

Abstract

We present a novel voice-based human-computer interface designed to enable individuals with motor impairments to use vocal parameters for continuous control tasks. Since discrete spoken commands are ill-suited to such tasks, our interface exploits a large set of continuous acoustic-phonetic parameters like pitch, loudness, vowel quality, etc. Their selection is optimized with respect to automatic recognizability, communication bandwidth, learnability, suitability, and ease of use. Parameters are extracted in real time, transformed via adaptation and acceleration, and converted into continuous control signals. This paper describes the basic engine, prototype applications (in particular, voice-based web browsing and a controlled trajectory-following task), and initial user studies confirming the feasibility of this technology.

1 Introduction

Many existing human-computer interfaces (e.g., mouse and keyboard, touch screens, pen tablets, etc.) are ill-suited to individuals with motor impairments. Specialized (and often expensive) human-computer interfaces that have been developed specifically for this target group include sip and puff switches, head mice, eye-gaze devices, chin joysticks and tongue switches. While many individuals with motor impairments have complete use of their vocal system, these devices make little use of it. Sip and puff switches, for example, have low communication bandwidth, making it impossible to achieve more complex control tasks.

Natural spoken language is often regarded as the obvious choice for a human-computer interface. However, despite significant research efforts in automatic speech recognition (ASR) (Huang et al., 2001), existing ASR systems are still not sufficiently robust to a wide variety of speaking conditions, noise, accented speakers, etc. ASR-based interfaces are therefore often abandoned by users after a short initial trial period. In addition, natural speech is optimal for communication between humans but sub-optimal for manipulating computers, windows-icons-mouse-pointer (WIMP) interfaces, or other electro-mechanical devices (such as a prosthetic robotic arm). Standard spoken language commands are useful for discrete but not for continuous operations. For example, in order to move a cursor from the bottom-left to the upper-right of a screen, the user might have to repeatedly utter "up" and "right" or "stop" and "go" after setting an initial trajectory and rate, which is quite inefficient. For these reasons, we are developing alternative and reusable voice-based assistive technology termed the "Vocal Joystick" (VJ).

2 The Vocal Joystick

The VJ approach has three main characteristics:
1) Continuous control parameters: Unlike standard speech recognition, the VJ engine exploits continuous vocal characteristics that go beyond simple sequences of discrete speech sounds (such as syllables or words) and include e.g., pitch, vowel quality, and loudness, which are then mapped to continuous con-

[*]This material is based on work supported by the National Science Foundation under grant IIS-0326382.

trol parameters.
2) Discrete vocal commands: Unlike natural speech, the VJ discrete input language is based on a pre-designed set of sounds. These sounds are selected with respect to acoustic discriminability (maximizing recognizer accuracy), pronounceability (reducing potential vocal strain), mnemonic characteristics (reducing cognitive load), robustness to environmental noise, and application appropriateness.
3) Reusable infrastructure: Our goal is not to create a single application but to provide a modular library that can be incorporated by developers into a variety of applications that can be controlled by voice. The VJ technology is not meant to replace standard ASR but to enhance and be compatible with it.

2.1 Vocal Characteristics

Three continuous vocal characteristics are extracted by the VJ engine: *energy*, *pitch*, and *vowel quality*, yielding four specifiable continuous degrees of freedom. The first of these, localized acoustic *energy*, is used for voice activity detection. In addition, it is normalized relative to the current vowel detected (see Section 3.3), and is used by our current VJ-WIMP application (Section 4) to control the velocity of cursor movements. For example, a loud voice causes a faster movement than does a quiet voice. The second parameter, *pitch*, is also extracted but is currently not mapped to any control dimension in the VJ-WIMP application but will be in the future. The third parameter is *vowel quality*. Unlike consonants, which are characterized by a greater degree of constriction in the vocal tract, vowels have much inherent signal energy and are therefore well-suited to environments where both high accuracy and noise-robustness are crucial. Vowels can be characterized using a 2-D space parameterized by F1 and F2, the first and second vocal-tract formants (resonant frequencies). We initially experimented with directly extracting F1/F2 and using them for directly specifying 2-D continuous control. While we have not ruled out the use of F1/F2 in the future, we have so far found that even the best F1/F2 detection algorithms available are not yet accurate enough for precise real-time specification of movement. Therefore, we classify vowels directly and map them onto the 2-D vowel space characterized by degree of constriction (i.e., tongue height) and tongue body position (Figure 1). In our VJ-WIMP application, we use

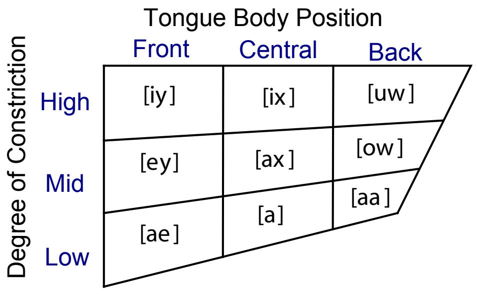

Figure 1: Vowel configurations as a function of their dominant articulatory configurations.

the four corners of this chart to map to the 4 principle directions of up, down, left, and right as shown in Figure 2 (note that the two figures are flipped and rotated with respect to each other). We have four different VJ systems running: A) a *4-class system* allowing only the specification of the 4 principle directions; B) a *5-class system* that also includes the phone [ax] to act as a carrier when wishing to vary only pitch and loudness; C) a *8-class* system that includes the four diagonal directions; and D) a *9-class* system that includes all phones and directions. Most of the discussion in this paper refers to the 4-class system.

A fourth vocal characteristic is also extracted by the VJ engine, namely *discrete sounds*. These sounds may correspond to button presses as on a mouse or joystick. The choice of sounds depends on the application and are chosen according to characteristic 2 above.

3 The VJ Engine

Our system-level design goals are modularity, low latency, and maximal computational efficiency. For this reason, we share common signal processing operations in multiple signal extraction modules, which yields real-time performance but leaves considerable computational headroom for the back-end applications being driven by the VJ engine.

Figure 3 shows the VJ engine architecture having three modules: signal processing, pattern recognition, and motion control.

3.1 Signal Processing

The goal of the signal processing module is to extract low-level acoustic features that can be used in

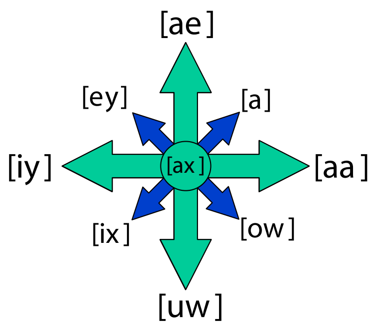

Figure 2: Vowel-direction mapping: vowels corresponding to directions.

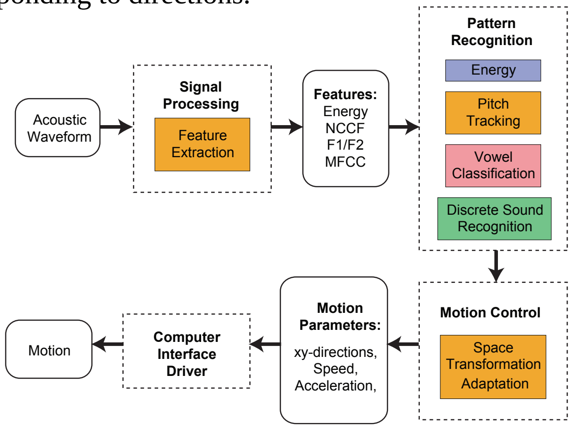
Acoustic Waveform
Signal Processing
Feature Extraction
Features: Energy NCCF F1/F2 MFCC
Pattern Recognition
Energy
Pitch Tracking
Vowel Classification
Discrete Sound Recognition
Motion Control
Space Transformation Adaptation
Motion Parameters: xy-directions, Speed, Acceleration,
Computer Interface Driver
Motion

Figure 3: System organization

estimating the vocal characteristics. The features we use are energy, normalized cross-correlation coefficients (NCCC), formant estimates, Mel-frequency cepstral coefficients (MFCCs), and formant estimates. To extract features, the speech signal is PCM sampled at a rate of F_s =16,000Hz. Energy is measured on a frame-by-frame basis with a frame size of 25ms and a frame step of 10ms. Pitch is extracted with a frame size of 40ms and a frame step of 10ms. Multiple pattern recognition tasks may share the same acoustic features: for example, energy and NCCCs are used for pitch tracking, and energy and MFCCs can be used in vowel classification and discrete sound recognition. Therefore, it is more efficient to decouple feature extraction from pattern recognition, as is shown in Figure 3.

3.2 Pattern Recognition

The pattern recognition module uses the acoustic features to extract desired parameters. The estimation and classification system must simultaneously perform *energy* computation (available from the input), *pitch tracking*, *vowel classification*, and *discrete sound recognition*.

Many state-of-the-art *pitch trackers* are based on dynamic programming (DP). This, however, often requires the meticulous design of local DP cost functions. The forms of these cost functions are usually empirically determined and/or their parameters are tuned by algorithms such as gradient descent (D.Talkin, 1995). Since different languages and applications may follow very different pitch transition patterns, the cost functions optimized for certain languages and applications may not be the most appropriate for others. Our VJ system utilizes a graphical model mechanism to automatically optimize the parameters of these cost functions, and has been shown to yield state-of-the-art performance (X.Li et al., 2004; J.Malkin et al., 2005).

For frame-by-frame *vowel classification*, our design constraints are the need for extremely low latency and low computational cost. Probability estimates for vowel classes thus need to be obtained as soon as possible after the vowel has been uttered or after any small change in voice quality has occurred. It is well known that models of vowel classification that incorporate temporal dynamics such as hidden Markov models (HMMs) can be quite accurate. However, the frame-by-frame latency requirements of VJ make HMMs unsuitable for vowel classification since HMMs estimate the likelihood of a model based on the entire utterance. An alternative is to utilize causal “HMM-filtering”, which computes likelihoods at every frame based on all frames seen so far. We have empirically found, however, that slightly non-causal and quite localized estimates of the vowel category probability is sufficient to achieve user satisfaction. Specifically, we obtain probability estimates of the form $p(V_t|X_{t-\tau},\dots,X_{t+\tau})$, where V is a vowel class, and $X_{t-\tau},\dots,X_{t+\tau}$ are feature frames within a length $2\tau+1$ window of features centered at time t. After several empirical trials, we decided on neural networks for vowel classification because of the availability of efficient discriminative training algorithms and their computational simplicity. Specifically we use a simple 2-layer multi-layer perceptron (Bishop, 1995) whose input layer consists of $26 * 7 = 182$ nodes, where 26 is the dimension of X_t, the MFCC feature vector, and $2\tau + 1 = 7$ is the number of consecutive frames, and that has 50 hidden nodes (the numbers 7 and 50 were determined empirically). The output layer has 4 output nodes representing 4 vowel probabilities. During training, the network is optimized to minimize the Kullback-Leibler (K-L) divergence between the output and the true label distribution, thus achieving the aforementioned probabilistic interpretation.

The VJ engine needs not only to detect that the user is specifying a vowel (for continuous control) but also a consonant-vowel-consonant (CVC) pattern (for discrete control) quickly and with a low probability of confusion (a VJ system also uses C and CV patterns for discrete commands). Requiring an initial consonant will phonetically distinguish these sounds from the pure vowel segments used for continuous control — the VJ system constantly monitors for changes that indicate the beginning of one of the discrete control commands. The vowel within the CV and CVC patterns, moreover, can help prevent background noise from being mis-classified as a discrete sound. Lastly, each such pattern currently requires an ending silence, so that the next command (a new discrete sound or continuous control vowel) can be accurately initiated. In all cases, a simple threshold-based rejection mechanism is used to reduce false positives.

To recognize the discrete control signals, HMMs are employed since, as in standard speech recognition, time warping is necessary to normalize for different signal durations corresponding to the same class. Specifically, we embed phone HMMs into "word" (C, CV, or CVC) HMMs. In this way, it is possible to train phone models using a training set that covers all possible phones, and then construct an application-specific discrete command vocabulary without retraining by recombining existing phone HMMs into new word HMMs. Therefore, each VJ-driven application can have its own appropriate discrete sound set.

3.3 Motion Control: Direction and Velocity

The VJ motion control module receives several pattern recognition parameters and processes them to produce output more appropriate for determining 2-D movement in the VJ-WIMP application.

Initial experiments suggested that using pitch to affect cursor velocity (Igarashi and Hughes, 2001) would be heavily constrained by an individual's vocal range. Giving priority to a more universal user-independent VJ system, we instead focused on relative energy. Our observation that users often became quiet when trying to move small amounts confirmed energy as a natural choice. Drastically different intrinsic average energy levels for each vowel, however, meant that comparing all sounds to a global average energy would create a large vowel-dependent bias. To overcome this, we distribute the energy per frame among the different vowels, in proportion to the probabilities output by the neural network, and track the average energy for each vowel independently. By splitting the power in this way, there is no effect when probabilities are close to 1, and we smooth out changes during vowel transitions when probabilities are more evenly distributed.

There are many possible options for determining velocity (a vector capturing both direction and speed magnitude) and "acceleration" (a function determining how the control-to-display ratio changes based on input parameters), and the different schemes have a large impact on user satisfaction. Unlike a standard mouse cursor, where the mapping is from 2-D hand movement to a 2-D screen, the VJ system maps from vocal-tract articulatory movement to a 2-D screen, and the transformation is not as straightforward. All values are for the current time frame t unless indicated otherwise. First, a raw direction value is calculated for each axis $j \in \{\mathrm{x}, \mathrm{y}\}$ as

$$d_j = \sum_i p_i \cdot \langle \mathbf{v}_i, \mathbf{e}_j \rangle \tag{1}$$

in which $p_i = p(V_t = i | X_{t-\tau,\dots,t+\tau})$ is the probability for vowel i at time t, $\mathbf{v}_i$ is a unit vector in the direction of vowel i, $\mathbf{e}_j$ is the unit-length positive directional basis vector along the j axis, and $\langle \mathbf{v}, \mathbf{e} \rangle$ is the projection of vector $\mathbf{v}$ onto unit vector $\mathbf{e}$. To determine movement speed, we first calculate a scalar for each axis j as

$$s_j = \sum_i \max\left[0, g_i\left(p_i \cdot f(\frac{E}{\mu_i})\right)\right] \cdot |\langle \mathbf{v}_i, \mathbf{e}_j \rangle|$$

where E is the energy in the current frame, μ_i is the average energy for vowel i, and $f(\cdot)$ and $g_i(\cdot)$ are functions used for energy normalization and perceptual scaling (such as logs and/or cube-roots). This therefore allocates frame energy to direction based on the vowel probabilities. Lastly, we calculate the velocity for axis j at the current frame as

$$V_j = \beta \cdot s_j^{\alpha} \cdot \exp(\gamma s_j). \tag{2}$$

where β represents the overall system sensitivity and the other values (α and γ) are warping constants, allowing the user to control the shape of the acceleration curve. Typically only one of α and γ is nonzero. Setting both to zero results in constant-speed movement along each axis, while $\alpha = 1$ and $\gamma = 0$ gives a linear mapping that will scale motion with energy but have no acceleration. The current user-independent system uses $\beta = 0.6$, $\gamma = 1.0$ and sets $\alpha = 0$. Lastly, the final velocity along axis j is $V_j d_j$. Future publications will report on systematic evaluations of different $f(\cdot)$ and $g_i(\cdot)$ functions.

3.4 Motion Control: User Adaptation

Since vowel quality is used for continuous control, inaccuracies can arise due to speaker variability owing to different speech loudness levels, vocal tract lengths, etc. Moreover, a vowel class articulated by one user might partially overlap in acoustic space with a different vowel class from another user. This imposes limitations on a purely user-independent vowel classifier. Differences in speaker loudness alone could cause significant unpredictability. To mitigate these problems, we have designed an adaptation procedure where each user is asked to pronounce four pre-defined vowel sounds, each lasting 2-3 seconds, at the beginning of a VJ session. We have investigated several novel adaptation strategies utilizing both neural networks and support vector machines (SVM). The fundamental idea behind them both is that an initial speaker-independent transformation of the space is learned using training data, and is represented by the first layer of a neural network. Adaptation data then is used to transform various parameters of the classifier (e.g., all or sub-portions of the neural network, or the parameters of the SVM). Further details of some of these novel adaptation strategies appear in (X.Li et al., 2005), and the remainder will appear in forthcoming publications. Also, the average energy values of each vowel for each user are recorded and used to normalize the speed control rate mentioned above. Preliminary evaluations on the data so far collected show very good results, with adaptation reducing the vowel classification error rate by 18% for the 4-class case, and 35% for the 8-class case. Moreover, informal studies have shown that users greatly prefer the VJ system after adaptation than before.

4 Applications and Videos

Our overall intent is for VJ to interface with a variety of applications, and our primary application so far has been to drive a standard WIMP interface with VJ controls, what we call the *VJ-WIMP* application. The current VJ version allows left button clicks (press and release, using the consonant [k]) as well as left button toggles (using consonant [ch]) to allow dragging. Since WIMP interfaces are so general, this allows us to indirectly control a plethora of different applications. Video demonstrations are available at the URL: `http://ssli.ee.washington.edu/vj`.

One of our key VJ applications is vocal web browsing. The video (dated 6/2005) shows examples of two web browsing tasks, one as an example of navigating the New York Times web site, the other using Google Maps to select and zoom in on a target area. Section 5 describes a preliminary evaluation on these tasks. We have also started using the VJ engine to control video games (third video example), have interfaced VJ with the Dasher system (Ward et al., 2000) (we call it the "Vocal Dasher"), and have also used VJ for figure drawing.

Several additional direct VJ-applications have also been developed. Specifically, we have directly interfaced the VJ system into a simple blocks world environment, where more precise object movement is possible than via the mouse driver. Specifically, this environment can draw arbitrary trajectories, and can precisely measure user fidelity when moving an object along a trajectory. Fidelity depends both on positional accuracy and task duration. One use of this environment shows the spatial direction corresponding to vocal effort (useful for training, forth video example). Another shows a simple robotic arm being controlled by VJ. We plan to use this environment to perform formal and precise user-performance studies in future work.

5 Preliminary User Study

We conducted a preliminary user study[1] to evaluate the feasibility of VJ and to obtain feedback regarding specific difficulties in using the VJ-WIMP system. While this study is not accurate in that: 1) it does not yet involve the intended target population

[1]The user study presented here used an earlier version of VJ than the current improved one described in the preceding pages.

of individuals with motor impairments, and: 2) the users had only a small amount of time to practice and become adept at using VJ, the study is still indicative of the VJ approach's overall viability as a novel voice-based human-computer interface method. The study quantitatively compares VJ performance with a standard desktop mouse, and provides qualitative measurement of the user's perception of the system.

5.1 Experiment Setup

We recruited seven participants ranging from age 22 to 26, none of whom had any motor impairment. Of the seven participants, two were female and five were male. All of them were graduate students in Computer Science, although none of them had previously heard of or used VJ. Four of the participants were native English speakers; the other three had an Asian language as their mother tongue.

We used a Dell Inspiron 9100 laptop with a 3.2 GHz Intel Pentium IV processor running the Fedora Core 2 operating system, with a 1280 x 800 24-bit color display. The laptop was equipped with an external Microsoft IntelliMouse connected through the USB port which was used for all of the tasks involving the mouse. A head-mounted Amanda NC-61 microphone was used as the audio input device, while the audio feedback from the laptop was output through the laptop speakers. The Firefox browser was used for all of the tasks, with the browser screen maximized such that the only portion of the screen which was not displaying the contents of the web page was the top navigation toolbar which was 30 pixels high.

5.2 Quantitative and Qualitative Evaluation

At the beginning of the quantitative evaluation, each participant was given a brief description of the VJ operations and was shown a demonstration of the system by a practiced experimenter. The participant was then guided through an adaptation process during which she/he was asked to pronounce the four directional vowels (Section 3.4). After adaptation, the participant was given several minutes to practice using a simple target clicking application. The quantitative portion of our evaluation followed a within-participant design. We exposed each participant to two experimental conditions which we refer to as input modalities: the *mouse* and the *VJ*. Each participant completed two tasks on each modality, with one trial per task.

The first task was a link navigation task (*Link*), in which the participants were asked to start from a specific web page and follow a particular set of links to reach a destination. Before the trial, the experimenter demonstrated the specified sequence of links to the participant by using the mouse and clicking at the appropriate links. The participant was also provided with a sheet of paper for their reference that listed the sequence of links that would lead them to the target. The web site we used was a Computer Science Department student guide and the task involved following six links with the space between each successive link including both horizontal and vertical components.

The second task was map navigation (*Map*), in which the participant was asked to navigate an online map application from a starting view (showing the entire USA) to get to a view showing a particular campus. The size of the map was 400x400 pixels, and the set of available navigation controls surrounding the map included ten discrete zoom level buttons, eight directional panning arrows, and a click inside the map causing the map to be centered and zoomed in by one level. Before the trial, a practiced experimenter demonstrated how to locate the campus map starting from the USA view to ensure they were familiar with the geography.

For each task, the participants performed one trial using the mouse, and one trial using a 4-class VJ. The trials were presented to the participants in a counterbalanced order. We recorded the completion time for each trial, as well as the number of *false positives* (system interprets a click when the user did not make a click sound), *missed recognitions* (the user makes a click sound but the system fails to recognize it as a click), and *user errors* (whenever the user clicks on an incorrect link). The recorded trial times include the time used by all of the above errors including recovery time.

After the completion of the quantitative evaluation, the participants were given a questionnaire which consisted of 14 questions related to the participants' perception of their experience using VJ such as the degree of satisfaction, frustration, and embarrassment. The answers were encoded on a 7-point Likert scale. We also included a space where the participants could write in any comments, and an in-

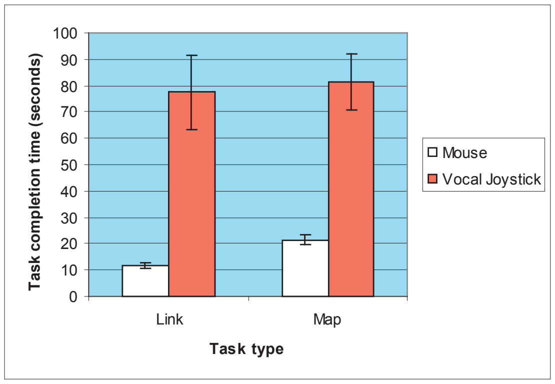

Figure 4: Task complement times

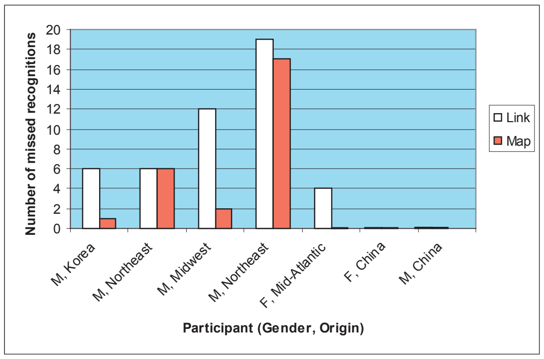

Figure 5: Missed recognitions by participant

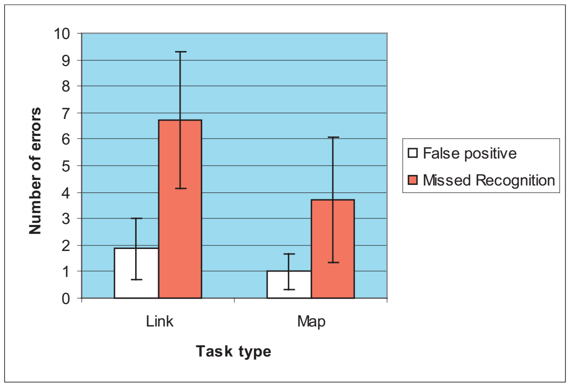

Figure 6: Average number of click errors per task

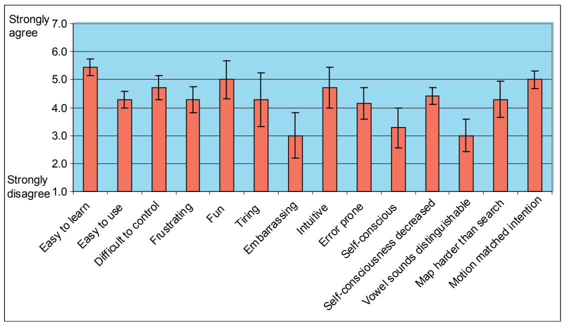

Figure 7: Questionnaire results

formal post-experiment interview was performed to solicit further feedback.

5.3 Results

Figure 4 shows the task completion times for Link and Map tasks, Figure 5 shows the breakdown of click errors by individual participants, Figure 6 shows the average number of false positive and missed recognition errors for each of the tasks. There was no instance of user error in any trial. Figure 7 shows the median of the responses to each of the fourteen questionnaire questions (error bars in each plot show $\pm$ standard error). In our measurement of the task completion times, we considered the VJ's recognition error rate as a fixed factor, and thus did not subtract the time spent during those errors from the task completion time.

There were several other interesting observations that were made throughout the study. We noticed that the participants who had the least trouble with missed recognitions for the clicking sound were either female or with an Asian language background, as shown in Figure 5. Our hypothesis regarding the better performance by female participants is that the original click sound was trained on one of our female researcher's voice. We plan also in future work to determine how the characteristics of different native language speakers influence VJ, and ultimately to correct for any bias.

All but one user explicitly expressed their confusion in distinguishing between the [ae] and [aa] vowels. Four of the seven participants independently stated that their performance would probably have been better if they had been able to practice longer, and did not attribute their perceived suboptimal performance to the quality of the VJ's recognition system. Several participants reported that they felt their vocal cords were strained due to having to produce a loud sound in order to get the cursor to move at the desired speed. We suspect this is due either to analog gain problems or to their adapted voice being too loud, and therefore the system calibrating the normal speed to correspond to the loud voice. We have since removed this problem by adjusting our adaptation strategy to express preference for a quiet voice.

In summary, the results from our study suggest that users without any prior experience were able to perform basic mouse based tasks using the Vocal Joystick system with relative slowdown of four to nine times compared to a conventional mouse. We anticipate that future planned improvements in the algorithms underlying the VJ engine (to improve accuracy, user-independence, adaptation, and speed) will further increase the VJ system's viability, and combined with practice could improve VJ enough so that it becomes a reasonable alternative compared to a standard mouse's performance.

6 Related Work

Related voice-based interface studies include (Igarashi and Hughes, 2001; Olwal and Feiner, 2005). Igarashi & Hughes presented a system where non-verbal voice features control a mouse system — their system requires a command-like discrete sound to determine direction before initiating a movement command, where pitch is used to control velocity. We have empirically found an energy-based mapping for velocity (as used in our VJ system) both more reliable (no pitch-tracking errors) and intuitive. Olwal & Feiner's system moves the mouse only after recognizing entire words. de Mauro's "voice mouse" `http://www.dii.unisi.it/~maggini/research/voice_mouse.html` focuses on continuous cursor movements similar to the VJ scenario; however, the voice mouse only starts moving after the vocalization has been completed leading to long latencies, and it is not easily portable to other applications. Lastly, the commercial dictation program Dragon by ScanSoft includes MouseGrid^{TM}(Dra, 2004) which allows discrete vocal commands to recursively 9-partition the screen, thus achieving log-command access to a particular screen point. A VJ system, by contrast, uses continuous aspects of the voice, has change latency (about 60ms) not much greater than reaction time, and allows the user to make instantaneous directional change using one's voice (e.g., a user can draw a "U" shape in one breath).

7 Conclusions

We have presented new voice-based assistive technology for continuous control tasks and have demonstrated an initial system implementation of this concept. An initial user study using a group of individuals from the non-target population confirmed the feasibility of this technology. We plan next to further improve our system by evaluating a number of novel pattern classification techniques to increase accuracy and user-independence, and to introduce additional vocal characteristics (possibilities include vibrato, degree of nasality, rate of change of any of the above as an independent parameter) to increase the available simultaneous degrees of freedom controllable via the voice. Moreover, we plan to develop algorithms to decouple unintended user correlations of these parameters, and to further advance both our adaptation and acceleration algorithms.

References

C. Bishop. 1995. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford.

2004. Dragon naturally speaking, Mousegrid^{TM}, ScanSoft Inc.

D.Talkin. 1995. A robust algorithm for pitch tracking (RAPT). In W.B.Kleign and K.K.Paliwal, editors, *Speech Coding and Synthesis*, pp. 495–515, Amsterdam. Elsevier Science.

X. Huang, A. Acero, and H.-W. Hon. 2001. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall.

T. Igarashi and J. F. Hughes. 2001. Voice as sound: Using non-verbal voice input for interactive control. In *ACM UIST 2001*, November.

J.Malkin, X.Li, and J.Bilmes. 2005. A graphical model for formant tracking. In *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*.

A. Olwal and S. Feiner. 2005. Interaction techniques using prosodic feature of speech and audio localization. In *Proceedings of the 10th International Conference on Intelligent User Interfaces*, pp. 284–286.

D. Ward, A. F. Blackwell, and D. C. MacKay. 2000. Dasher - a data entry interface using continuous gestures and language models. In *ACM UIST 2000*.

X.Li, J.Malkin, and J.Bilmes. 2004. A graphical model approach to pitch tracking. In *Proc. Int. Conf. on Spoken Language Processing*.

X.Li, J.Bilmes, and J.Malkin. 2005. Maximum margin learning and adaptation of MLP classifers. In *9th European Conference on Speech Communication and Technology (Eurospeech'05)*, Lisbon, Portugal, September.