Behavior Research Methods, Instruments, & Computers
1994, 26 (2), 260-265

12. SIMULATION MODELING

Chaired by Barbara Basden, *California State University at Fresno*

Development and experimentation with synthetic visible speech

MICHAEL M. COHEN and DOMINIC W. MASSARO
University of California, Santa Cruz, California

We have implemented software for development of synthetic visual speech and perceptual experimentation on a UNIX workstation. We describe recent improvements in the speech synthesis and the capabilities of the development system. We also show how a typical experiment is programmed and describe our solutions for real-time experimental control under the UNIX operating system.

Speech perception and speech production are skills that rival other impressive human achievements. Even after decades of intense effort, speech recognition by machine remains far inferior to that of human performance. Also, as we have all experienced, speech synthesis has not yet obtained the naturalness and quality of the speech of even a young child. Our research has led us to the view that speech perception is easy for human beings because multiple sources of information support the identification and interpretation of the language input. Therefore, it is important to define the sources of information and to determine how they are evaluated and integrated to achieve recognition. The paradigm that we have developed permits us to determine which of the many potentially functional cues are actually used by human observers. The systematic variation of properties of the speech signal, combined with quantitative tests of models based on different sources of information, enables the investigator to test the psychological validity of different cues. Thus, our research strategy not only addresses how different sources of information are evaluated and integrated, it can uncover what sources of information are actually used. We believe that the research paradigm confronts both the important psychophysical question of the nature of information and the process question of how the information is transformed and mapped into behavior.

Valuable and effective information is afforded by a view of the speaker's face in speech perception and recognition by humans. Visible speech is particularly effective when the auditory speech is degraded by noise, bandwidth filtering, or hearing impairment. As an example, the perception of sentences that have been bandpass filtered or distorted by auditory noise improves dramatically when subjects are permitted a view of the speaker (Massaro, 1987). Such improvement has also been observed in hearing-impaired listeners and patients with cochlear implants (Massaro, 1987). The strong influence of visible speech is not limited to situations with degraded auditory input, however. A perceiver's recognition of an auditory-visual syllable reflects the contribution of both sound and sight. If an auditory syllable /ba/ is dubbed onto a videotape of a speaker saying /da/, subjects often perceive the speaker to be saying /ða/ (Massaro, 1987, 1989; Massaro & Cohen, 1990).

A main objective of our research is to identify the facial properties that are informative by evaluating the effectiveness of various properties in a synthetic animated face. Analogous to the valuable contribution of using auditory speech synthesis in speech perception research, visible speech synthesis permits the type of experimentation necessary to determine (1) what properties of visible speech are used, (2) how they are processed, and (3) how this information is integrated with auditory information and other contextual sources of information in speech perception. This experimental and theoretical framework has already established several facts concerning speech by eye and ear.

The continued development and utilization of a realistic, high-quality, facial display has provided a powerful tool for investigating a number of questions in auditory-visual speech perception. The analysis of the articulation of real speakers has guided the development of the visible speech synthesis. In addition, perception experiments have indicated how well the synthesis simulates real speakers. At the same time, a better understanding of visible speech

The research reported in this paper and the writing of the paper were supported, in part, by Public Health Service Grant PHS R01 DC 00236 to D.W.M. Correspondence should be addressed to M. M. Cohen, Program in Experimental Psychology, University of California, Santa Cruz, CA 95064 (e-mail: mmcohen@fuzzy.ucsc.edu).

perception derived from perceptual studies has assisted in the development of the display.

This paper updates our prior work (Cohen & Massaro, 1990; Massaro & Cohen, 1990) on the synthesis and use of visual speech. The reader is referred to these reports and also to Cohen and Massaro (1993) for surveys of earlier work on visual speech synthesis.

Development Environment

Our current synthesis software is a direct descendant of Parke's (1974, 1975, 1982) parametrically controlled polygon topology synthesis technique, incorporating code developed by Pearce, Wyvill, Wyvill, and Hill (1986) and ourselves (Cohen & Massaro, 1990; Massaro & Cohen, 1990). Our facial synthesis and perceptual experimentation is carried out on a Silicon Graphics, Inc. (SGI) 4D/CRIMSON-VGX workstation under the IRIX (SGI SYS V UNIX) operating system. This machine has a speed of roughly 85 MIPS, 16 MFLOPS, and 1,000,000 polygons displayed per second. The software consists of roughly 12,000 lines of C code and uses the SGI graphics library (GL) calls and Overmars's (1990) Forms Library to construct the graphical user interface (GUI) for development. Compared with earlier hardware (SGI-3030), which produced the animation to tape frame-by-frame at a rate of roughly 1 min per frame, the current hardware can produce 60 frames per second, which allows real-time manipulation of parameters and real-time production of stimuli for perceptual experiments. A smaller version of the visual speech software with the same functionality but without the GUI is available for use under f77 main programs for perceptual experiments, including the presentation of auditory speech and collection of responses from human participants. A typical experiment will be described in the section Experimental Control Software.

Figure 1 shows the GUI for face development. The master panel in the lower right of the screen has facial controls, facilities for editing speech segment definitions, sentence input, speaking rate, parameter tracking, call-ups for subsidiary control panels, and other miscellaneous controls. The upper right panel is a text-based interface that allows one to control the face by using alpha-

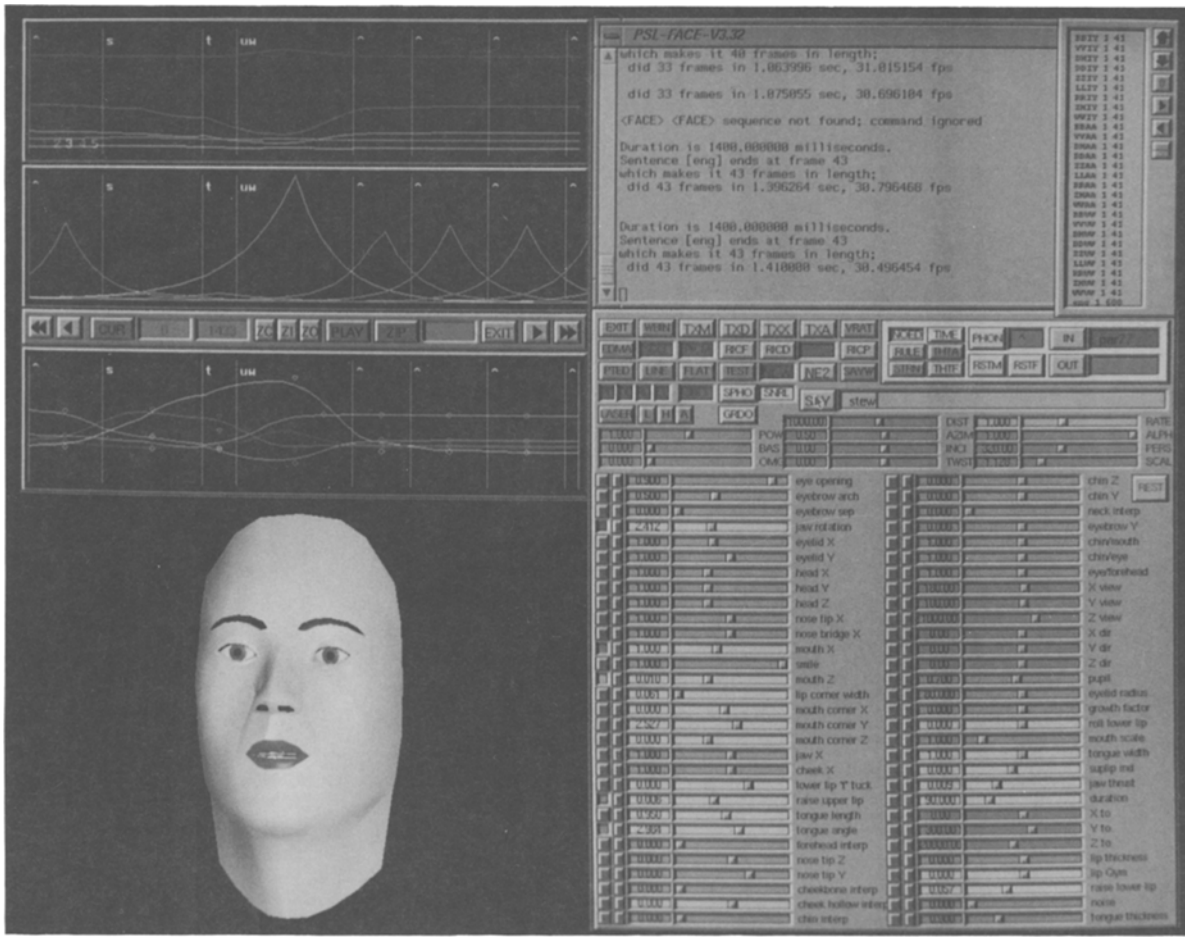

Figure 1. Graphical user interface for face development. Master panel at lower right has facial controls, facilities for editing speech segment definitions, sentence input, speaking rate, parameter tracking, call-ups for subsidiary control panels, and other miscellaneous controls. Upper right panel is text interface. Lower left panel is display output. Upper left comprises play control with cursors for zooming and moving face in time, and plots of control parameters (bottom), dominance functions (middle), and derived lip measures (top).

numeric commands or files of such commands. Most graphically controllable functions can also be specified by such commands, along with some additional commands (e.g., for sequencing of expression) that are not yet controlled graphically. Also in the upper right of the screen is a menu panel for the selection of members of a set of tokens for synthesis. In this example, the menu is set to call one of 27 consonant-vowel syllables whose definitions have been read in from a file. The lower left panel is the display output. This area can also be output in NTSC video by using the SGI broadcast video output option. The upper left area contains the play controls with cursors for temporal zooming and displaying the face forward and backward in time, and plots of control parameters (bottom), dominance functions (middle), and derived facial measures (top). Alternate displays for the top position (not shown in the figure) include plots of digitized audio signals and speech spectrograms. The controls in this region also allow modification of segment durations, which is especially useful for synchronization with natural speech.

The facial model has a tongue implemented as a shaded surface made of a polygon mesh, controlled by several parameters: tongue length, angle, width, and thickness. Figure 2 shows a side view of the face with a protruding tongue.

Coarticulation in Synthetic Visual Speech

An important improvement in our visual speech synthesis software has been the development of a new algorithm for articulator control that takes into account the phenomenon of coarticulation (Cohen & Massaro, 1993). *Coarticulation* refers to changes in the articulation of a speech segment depending on preceding (*backward* coarticulation) and upcoming (*forward* coarticulation) segments. An example of backward coarticulation is the difference in articulation of a final consonant in a word depending on the preceding vowel, as in *boot* versus *beet*. An example of forward coarticulation is the anticipatory lip rounding at the beginning of the word *stew*. Great improvement of more recent auditory speech synthesizers, such as MITtalk (Allen, Hunnicutt, & Klatt, 1987) and DECtalk (1985), over the previous generation of synthesizers such as VOTRAX (1981), is partly due to the inclusion of rules specifying coarticulation among neighboring phonemes.

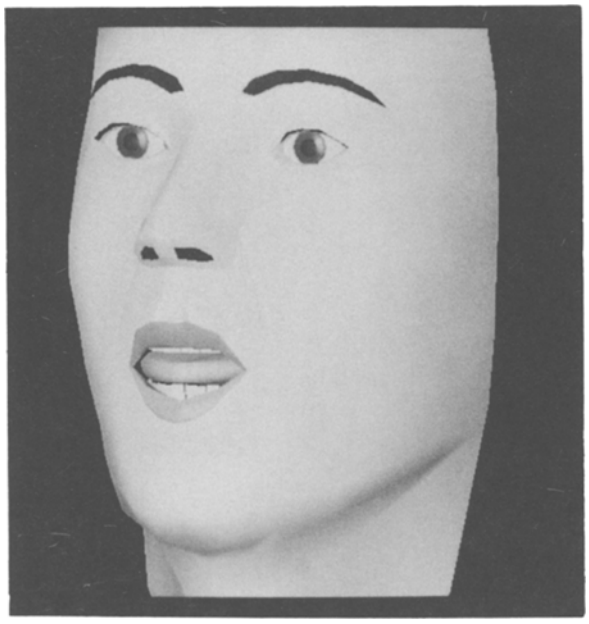

Figure 2. Side view of face with tongue.

Our approach to the synthesis of coarticulated speech is based on the articulatory gesture model of Löfqvist (1990). The speech segment has dominance over the vocal articulators that increases and then decreases over time during articulation. Adjacent segments will have overlapping dominance functions, which leads to a blending over time of the articulatory commands related to these segments. Given that articulation of a segment is implemented by several articulators, there is a dominance function for each articulator. The different articulatory dominance functions can differ in time offset, duration, and magnitude.

An example of the system's operation is shown in the top panel of Figure 3, illustrating the lip protrusion dominance functions for the word *stew*. As can be seen, the /s/ and /t/ segments, in comparison with /u/, have very low dominance with respect to lip protrusion. Also, the dominance of /u/ extends far forward in time. The middle panel gives the resulting lip protrusion trace. One can see how the lip protrusion extends forward in time from the vowel. Note that the figure only illustrates the dynamics for lip protrusion. Other control parameters, such as tongue angle, for /t/ and /u/ have equal dominance. This allows the tongue to reach its proper location against the back of the upper teeth for /t/. The bottom panel of the figure shows a side view of the face during the /s/ segment at the time indicated by the vertical red cursor in the top two panels.

Experimental Control Software

We will now describe how perceptual experiments with synthetic visual speech and other stimuli are carried out on our UNIX workstation. We will use as an example a typical factorial design with five levels of synthetic visual speech crossed with five levels of synthetic auditory speech. In both cases, the five levels consist of a continuum spanning the range between the syllables /ba/ and /da/. The task of the subject is to identify what is said on each trial. Figure 4 shows the f77 code for experimental control. Most experimental activities are carried out by calls to a library of f77 and C subroutines. In this example, we start with arrays specifying the independent variables *v(5)* and *a(5)*. The graphics hardware and facial synthesis software are initialized with a call to *faceinit*, which also sets the viewpoint. The following three calls set the facial synthesis segment parameter definitions, the visual /b/-/d/ segment continuum, and the list of /ba/-/da/ visual tokens. The next step in the experiment is to initialize the audio hardware (VIGRA MMI-210) and read

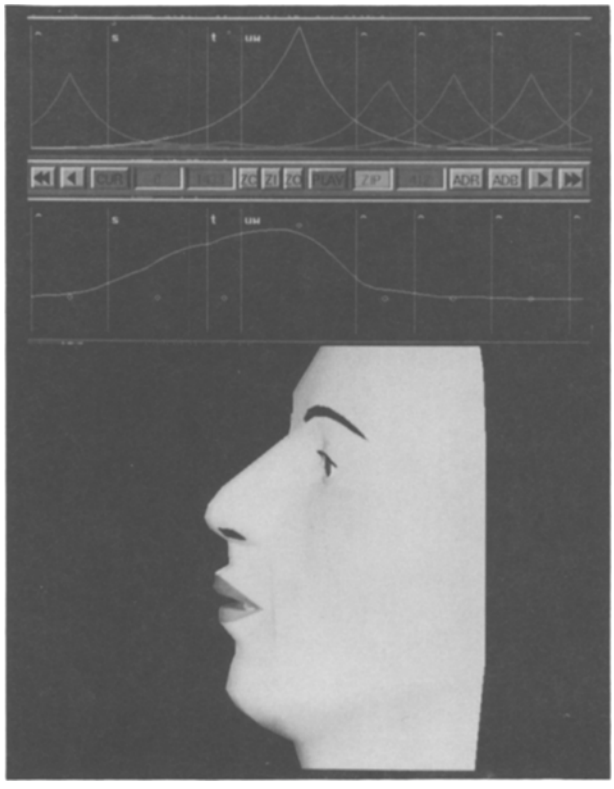

Figure 3. Dominance functions (top panel) and parameter control functions (middle panel) for lip protrusion for the word *stew*. Bottom panel shows facial display during the /s/ segment.

in the five speech sounds from disk files to memory. Once loaded, the sounds are available for immediate presentation, and they can be referred to simply by sound number according to loading order.

The next call, *expsgi*, alters the state of the computer for experimental use and initializes the experimental control software, specifying the code name of the experiment, the response collection method, response key mapping (or other secondary characteristic of the responses), and the independent variables and number of levels involved. Responses are made on video terminals (TVI-950) connected with the computer by asynchronous lines. Many response forms are available, including: (1) unmapped single keys, (2) mapped single keys, (3) multiple-key text, and (4) response scales shown as bars on terminal with moving cursor indicator. For all methods, reaction times for each keypress are automatically collected. In the present example, mapped keyboard responses are used, with the "b" and "d" keys being translated to "1" and "2" in the data files. The design used here specifies 5 levels of visual stimuli and 5 levels of auditory stimuli, completely crossed and sampled without replacement in blocks of 25 stimulus combinations. Negative numbers for the levels would indicate sampling with replacement, and a number of other variations can be specified. When the *expsgi* call is executed, the experimenter is asked via the console to enter information concerning randomization initialization, group and run number, and the code numbers for 1–4 subjects being tested together (in separate sound- and light-controlled rooms. The system automatically maintains a logfile for each experiment, keeping track of when each run has occurred, the conditions, subjects, data filenames, and so on.

Following the *expsgi* call, the computer graphics hardware is set to output NTSC video using a *system* call which passes commands to the operating system. The timer starts counting out 2,000 msec until the beginning of the first trial using the *timers* call. Several types of timing routines are available, including *timer*, which waits the specified number of milliseconds before returning, and *timers*, which just starts an interval. Later, one calls *twait* to wait for the conclusion of the specified interval.

The next routine called is *sess*, which here specifies that a session should be run using the *trial* subroutine for each trial. In the current example, 10 practice trials are followed by 250 recorded trials, and the independent variables (IVs) are *v* and *a*. Following the experimental session, *endrun* is called to wrap up the data files and restore the computer to normal operations.

The *trial* routine specifies what happens on each trial. The first statement repeats the specification of the IVs. In the example, we then form a command string *rs* for the rendering of the audio-video stimuli. Of special interest are the function calls to *ivn* that return the level of the IVs selected for the current trial. Another useful function is *iv*, which returns the *value* of the IV (i.e., the content of the selected level of the particular IV array) for the current trial. Then, the *twait* call waits for the intertrial interval (ITI) to finish.

At this point subject responses are enabled, and the animated audio-video speech stimulus is presented using *facerend* over color monitors. There are many other routines for stimulus presentation—alphanumeric displays on the subject display terminals, nonspeech auditory signals with amplitude control, arbitrary visual displays (e.g., pictures) from disk files, graphics done on the fly, and video from external sources such as computer-controlled videotape or laser disk. A Panasonic MX-50 audio-video switcher allows selection of the various stimulus sources under serial line control from the computer.

Following presentation of the stimulus, *conrspw* is called, which waits for all subjects to respond with a valid key before going on, with a warning bell sent to any subject taking more than 8,000 msec. Other routines are available without the warning tone and for fixed response intervals. Following the response interval, *timers* is called to start the next ITI and then control is returned to *sess* for processing of the responses and setting IV levels for the following trial.

Real-Time Experimental Control Under IRIX: Problems and Solutions

Using a multiprocessing operating system like IRIX for running experiments is fraught with difficulties, especially for tasks that require heavy computation, such as computer animation. Even with the SGI's parallel graphics engines, considerable work must be done by the CPU to

```
      integer v(5),a(5)                  ! independent variables
      character*8 fn                     ! file name for audio
      external trial

      call faceinit(180.,100.,1000.)     ! initialize face display
      call faceget('prm87')              ! face parameter definitions
      call faceget('bd5.con')            ! visual continuum definition
      call faceget('bd5.seg')            ! visual token list

      call sinit                         ! initialize audio
      do 10 j=1,5
        write(fn,"('dmv'i1'.kwv')")j
10      i = sopen(fn)                    ! load audio files

c      specify experiment,response method,resp mapping, IV's
      call expsgi('pg0 : sgi proto',4,'/b/d/',v,5,0,a,5,0)
      call system('setmon -n -g NTSC') ! set display for video out
      call timers(2000)                  ! start timer for 2000 msec
      call sess(trial,10,250,v,a)        ! run a session of trials
c                  10 practice, 250 recorded
      call endrun                        ! wrap things up
      stop
      end

      subroutine trial(v,a)              ! specification of trial events
      integer v(5),a(5)                  ! independent variables
      character*28 rs
c       write string rs specifying visual and auditory events
c       based on random iv values selected for this trial
c                    v token,frames,a token,start point,loudness
      write(rs,"('render bd'i1' 1 26 'i2' 467. 4095')")ivn(v),ivn(a)
      call twait                         ! wait for ITI to be done
      call enable                        ! enable subject responses
      call facerend(rs)                  ! display the A & V syllables
      call conrspw(0,8000)               ! collect reponses (warn after 8sec)
      call timers(1000)                  ! start next ITI of 1000 msec
      return                             ! done with this trial
      end
```

Figure 4. Typical experimental control program, described in the text.

achieve real-time synthetic visual speech output. There may be other users and other processes on the machine; network accesses may occur such as telnet, rlogin, ftp, ping, finger, file sharing such as NFS may be occurring; mail arrives; and the control program might be swapped out. Some of these problems can be avoided by policy—for example, if the experimental machine is not involved in file sharing and other users are asked to log out before experimental use begins.

Under IRIX, activation of network demons (programs that carry out the network services like ftp and rlogin) is controlled by a configuration file */usr/etc/inetd.conf* that lists which services should be allowed. Although this file and the *inetd* program that starts the demons are ordinarily protected from modification by users, we have constructed special programs that allow an ordinary user to modify the operating environment. To prevent network access during the experiments, a short program, *netoff*, is called from *expsgi*. This program is owned by ROOT, and when executed by an ordinary user, it takes on the ROOT ability to control the system. After it has taken control, a special experimental version of the UNIX network configuration file *inetd.conf* is copied into place, the *inetd* program is restarted, and mail is stopped. The experimental version of the control file has demons such as ftp, telnet, and so forth, disabled. The program *neton* reverses the process, restoring normal network operations.

In the timing of experimental events, the system timer call *gettimeofday* is used as the basis of measurements. Ordinarily, the system lock resolution is set to 100 Hz, but a faster rate (1200 Hz) option has been set in the system configuration files, which gives 833-μsec time resolution with a slight reduction in overall system speed.

Normally under IRIX, various processes are given control of the CPU in rotation according to certain degradable priorities and availability of needed resources. If a process is waiting for input, it might be suspended in favor of another process that is ready to go. This procedure

could cause difficulties when the program is waiting for a subject's response, because the code for processing the response might not be running. To ensure that other processes do not cause execution of the experimental program to be preempted, the system routine *schedctl* is called by *expsgi* to set the highest possible nondegrading priority. When this is set, only emergency and error conditions in the system are given higher priority.

One of the most difficult problems in a system like IRIX is how to collect responses and measure latencies with sufficient (millisecond) accuracy. Often, serial line drivers will collect a buffer full of characters before informing a calling program of their arrival. One solution to this problem is to write one's own device driver and interrupt service routine. For simpler single-user systems such as a PDP-11 or PC, this is relatively easy to do in machine language, but for large UNIX systems, the interrupt and communications hardware is not well documented and the integration of new drivers is a quite difficult task. Our solution, which achieves a response latency of about 400 μsec in accepting responses, consists of two parts. First, with *ioctl* calls, the serial lines are configured to use no buffering and no waiting before reporting a received character. We also set the *SIGPOLL* signal to occur on character receipt, and dispatch control in that event to our own signal handling routine, which operates like an interrupt routine.

Retrospective

The applied value of visible speech is its potential to supplement other (degraded) sources of information for disabled individuals. Its use is important for hearing-impaired individuals, because it allows effective communication within spoken language—the universal language of the community. Just as synthetic auditory speech has been of great importance for research on auditory speech perception, synthetic visual speech is important in the study of visual speech perception. In addition, just as auditory speech synthesis has proved a boon to our visually impaired citizens in human machine interaction, visual speech synthesis may prove to be valuable for the hearing impaired. As just one example, cochlear implants have proved successful in allowing implanted individuals to communicate via spoken language. In almost all of the cases, however, the auditory speech has not been adequate by itself, and visible speech has been shown to provide an important source of information for successful communication. Visible speech synthesis provides a medium for presenting this channel of information.

Animated visible speech, like synthetic audible speech, has the potential of being communicated over low bandwidth channels. Parametric synthesis means that only the parameters need be transmitted from the sender to the receiver. These parameters could then be used to synthesize the message locally at the receiver's station. If this local synthesis is not possible, stylized representations of the visible speech could be sent over telephone lines or other low-bandwidth channels. In the near future, much of telecommunications will involve a global society communicating by ear and eye.

Although the supporting software for UNIX-based workstations is more complicated than that for PCs, researchers on perception should still be able to write fairly simple experimental programs. As the cost of these powerful machines continues to decline, we believe that more psychologists will shift to their use.

REFERENCES

ALLEN, J., HUNNICUTT, M. S., & KLATT, D. (1987). *From text to speech: The MITalk system.* Cambridge, MA: Cambridge University Press.

COHEN, M. M., & MASSARO, D. W. (1990). Synthesis of visible speech. *Behavior Research Methods, Instruments, & Computers*, **22**, 260-263.

COHEN, M. M., & MASSARO, D. W. (1993). *Modeling coarticulation in synthetic visual speech.* In N. M. Thalmann & D. Thalmann (Eds.), *Models and techniques in computer animation* (pp. 139-156). Tokyo: Springer-Verlag.

DECTALK (1985). *Programmers reference manual.* Maynard, MA: Digital Equipment Corporation.

LÖFQVIST, A. (1990). Speech as audible gestures. In W. J. Hardcastle & A. Marchal (Eds.), *Speech production and speech modeling* (pp. 289-322). Dordrecht: Kluwer Academic Publishers.

MASSARO, D. W. (1987). *Speech perception by ear and eye: A paradigm for psychological inquiry.* Hillsdale, NJ: Erlbaum.

MASSARO, D. W. (1989). [Précis of *Speech perception by ear and eye: A paradigm for psychological inquiry*]. *Behavioral & Brain Sciences*, **12**, 741-794.

MASSARO, D. W., & COHEN, M. M. (1990). Perception of synthesized audible and visible speech. *Psychological Science*, **1**, 55-63.

OVERMARS, M. (1990). *Forms library.* Utrecht: Utrecht University, Department of Computer Science.

PARKE, F. I. (1974). *A parametric model for human faces* (Tech. Rep. No. UTEC-CSc-75-047). Salt Lake City: University of Utah, Computer Science Department.

PARKE, F. I. (1975). A model for human faces that allows speech synchronized animation. *Journal of Computers & Graphics*, **1**, 1-4.

PARKE, F. I. (1982). Parameterized models for facial animation. *IEEE Computer Graphics*, **2**, 61-68.

PEARCE, A., WYVILL, B., WYVILL, G., & HILL, D. (1986). Speech and expression: A computer solution to face animation. In *Proceedings of Graphics Interface '86* (pp. 136-140).

VOTRAX (1981). *User's manual.* Troy, MI: Votrax, Div. of Federal Screw Works.