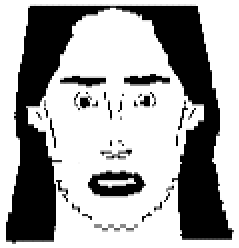
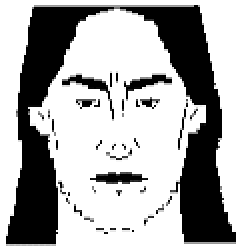
MICRO - The 6502/6809 Journal
No. 49, June, 1982

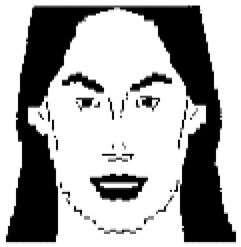
Face Synthesizer for PET

by David Heise

This program creates an animated face on a PET screen that changes expression on keyboard command. Animation is controlled from BASIC programs, so the PET face can be used in any application-education, marketing, games, etc.







Face Synthesizer
requires:
PET
8K or larger
3.0 or 4.0 Operating System

Facial expressions reflect emotions. This well-known fact has been a topic of study for social psychologists, who analyze expressions by observing the shape and position of eyes, mouth, eyebrows, and other facial features. The psychologists' precise technical descriptions of these features provided me with the data I needed to represent facial expressions on a computer. Now your PET can express emotions too.

This program generates a face that smiles, winks, and pouts; shows fear, disgust, or anger, or widens its eyes in surprise. You operate the face from the keyboard. The facial expressions also can be called from BASIC programs for practical applications. A salesroom display program could use the PET face to call attention to a product with moving eyes, winks, and smiles. Education programs could use the face to provide rewards or reproofs for right or wrong answers.

What Are Facial Expressions?

Emotional messages are constructed on the face by the shape of the mouth, eyes, and eyebrows (and sometimes the nose, cheeks, and forehead as well). Each of these features has a limited number of major shapes produced by the action of certain facial muscles. Whether a group of muscles is tugging gently or straining hard may suggest the intensity of feeling, but the real information is in the fact that certain muscles are operative, producing the characteristic shape for that muscle group.

The brows have four major shapes other than a neutral relaxed position. They may be curved upward (as in surprise), flattened and raised (as in fear), flattened and lowered (as in sadness), or pulled down and inward (as in anger).

The opened eyes have six major shapes: neutral, wide open (as in surprise), raised lower lids (as in disgust), raised and tensed lower lids (as in fear), squinting (as in anger), and upper lids drooping and sloped (as in sadness).

Major shapes of the mouth, aside from neutral, are: dropped open (as in surprise), corners pulled horizontal (as in fear), lips pressed tight (as in anger), squared outthrust lips baring teeth (as in anger), upper lip pulled up (as in disgust), corners down (as in sadness), corners raised (as in happiness, with extra stretching for smiles, grins, or laughs).

The end of the nose may be normal or raised by pressure from the upper lip; the upper nose may be normal or crinkled. Cheeks may be normal or raised during laughter. The forehead may be normal or wrinkled by pressures from the eyebrows.

Variations in one feature combine with variations in another feature; for example, any eyebrow formation can occur with any mouth shape. But not quite every combination of features is possible. For example, the mouth isn't disgusted alone; "disgusted" mouth occurs with nose raised.

Expressions for the primary emotions are universal. Surprise combines arched eyebrows with wide open eyes and a dropped open mouth. Fear shows in raised and flattened eyebrows, raised and tensed lower eyelids, along with side stretched lips. Disgust involves raised lower eyelids, and the upper lip curled up so to raise the nose; the upper nose may be crinkled. In anger the brows pull down and inward, the eyes squint, and the lips either are pressed tight or squared into a snarl. Happiness is revealed in upturned corners of the mouth; laughing also raises the cheeks which m turn may push the lower eyelids up.

Blends can be formed by combining signs of two emotions. For example, arched eyebrows and a smile indicate surprised happiness. Subtle feelings also may be communicated by rapid sequences of expressions - an angry expression interrupted by a flash of disgust.

The Face Program

The face synthesizer presented here consists of 2K of assembly language code designed to run on 32K PET/CBM microcomputers with 40-column screens. Instructions are given on relocating the code for 16K or 8K machines. The synthesizer does not work with Commodore operating system 1.0, and it produces a long and narrow face on 80-column screens.

The facial image was created by tracing a photo of a woman's face in a magazine onto graph paper, and then matching features in each cell as closely as possible with Commodore graphics. The happy, grinning face that appears by default is the original. Feature variations were' treated artistically, with guidance from photographs of facial expressions. Gaze variations were constructed so that the face can be made to look forward, left, right, or down. Left and right eyelids can be independently controlled for winks, blinks, and closed eyes.

The program includes limited feature variations. Each feature shape is represented in a single form, though

. *End p. 31; begin p. 32.*

real faces can produce gradations. Some eye variations are too subtle for Commodore graphics, so a single approximate shape has to serve multiple duty. Nose crinkling for disgust and raised cheeks for laughter are not included. No asymmetric moves for brows or mouth are included.

Paul Ekman and Wallace Friesen's book, *Unmasking the Face* (Prentice Hall, 1975), guided composition of the feature variations. This paperback is an essential manual for anyone using the face program in application programming.

The Emotional Keyboard

Running the programs listed here produces the PET face and an initial display of some of its expressive variations. After the opening show, you can produce new expressions from the keyboard. Each feature variation is linked to a single key. The brow is controlled by keys in the top row of the keyboard. Eyes and eyelids are controlled by keys in the next two rows, with left eye variations on the left and right eye variations on the right. The mouth is controlled by keys in the bottom row. The six basic emotion configurations are available from single keys - parentheses, brackets, and inequalities. The number keys control compositions that are needed in programming. Chart 1 indexes command characters and their effects.

The demonstration program allows you to construct a string of commands that can produce an animated sequence of expressions. To begin a string, press the "+" key. When you have completed the sequence and are ready to view it, press the "." key. Use the delay command (shifted SPACE) between different expressions so that the sequence runs slow enough for you to see. You may press the "." key to see the last sequence again. If you press the " /" key, the commands producing the last sequence are printed on the screen and the program ends. Use this option to work out desired effects before entering commands into your BASIC program.

Listings of computer code and data are omitted from this World Wide Web reproduction of the article.
The text after this point is concerned with programming issues.

Faces in BASIC

Listing 1 presents the demonstration program that illustrates all of the points essential to using the Face routine with BASIC. Commands are sent to the routine via a string variable named FACE$ (or FA$) (this must be defined in the program before calling the Face routine or you will get an error). The FACE$ string may be constructed by direct quotes; by GET, INPUT, or READ statements; or by string manipulations.

The first string sent to the routine should begin with "01234". These five commands display the face on the screen. Thereafter the FACES string consists only of commands for desired feature changes.

Commands in the FACES string are implemented by SYS 30729 on a 32K PET, SYS 14345 on a 16K PET, or SYS 6153 on an 8K PET.

Listing 2 is a BASIC utility program that automatically loads the Face code from a file named CODE, protects the code from BASIC, and then loads another BASIC program named MAIN. Program MAIN would be listing 1 when you set up the demonstration procedure; otherwise it is the application program you have written. On tape, the Loader program should be first, the CODE file second, and MAIN third. On disk, the order is immaterial, but the names CODE and MAIN are required. (The DOS wedge can be loaded after the face routines are 1oaded.)

Assembly-Language Routine

Listing 3 is the assembly-language routine. Listing 4 contains the data used to compose the face and its variations on the screen.

You enter the code in listings 3 and 4 with the PET/CBM monitor (SYS 1024). To begin enter:

```
    .M 7800,787F
```

Then overwrite the contents of the cells with the hexadecimal values at the left of listing 3, pressing RETURN after each line. Continue with:

```
    .M 7880,7900
```

and so on. When you have finished

. *End p. 32; begin p. 33.*

with the data in listing 3, go on to listing 4. Print out lines of memory the same as they appear in listing 4 to simplify entry.

When all code has been entered, save it on tape with:

```
    .S "CODE",01,7800,7FF2
```

or on a preinitialized disk in unit 0 with:

```
    .S "O:CODE",08.7800,7FF2
```

If you are saving on tape, remember to save the Loader program (listing 2) on the tape before you begin to enter the code in listings 3 and 4.

To relocate the code for a 16K machine, change all $7000 addresses to $3000. For example, you would begin entering code with:

```
    .M 3800,387F
```

Some addresses within the program have to be changed. Relevant lines are flagged by < SIZE > in the comments column. Change the 7 in the address high byte to 3 wherever < SIZE> appears. For example, A9 77 in line 440 would become A9 37. In addition, the last byte of each entry in the INDEX must be changed (the INDEX begins at line 2080). For example, 79 in the SCALP entry would be changed to 39, 7A in the BROW entry becomes 3A, etc.

Relocation for 8K is similar except that sevens are changed to ones.

The address in line 1890 of listing 3 is for operating system 3.0. Change 69 C3 to 00 BF for operating system 4.0.

The data in listing 4 remain the same for all machines.

Program Notes

Lines 70-100 in listing 3 are instructions for the assembler program.

Lines 140-210 indicate parts of the PET/CBM operating system that are used in the Face program.

Lines 240-350 show the locations in screen memory where facial features begin.

Lines 420-460 are a short routine for putting the Face code outside the bounds of BASIC. This routine is called by the Loader program in listing 2 immediately after loading the Face code.

The Face program begins in line 540. First the FACE$ command string has to be found (lines 540-590 plus the subroutine in lines 1630-2030).

Commands in the FACE$ string are transferred to the stack in reverse order (lines 620-670), after a zero is pushed on the stack to signal the end of the commands (lines 600-610).

Commands are retrieved from the stack by the routine in lines 700-740. If a zero is encountered, then all commands have been processed and control returns to BASIC. If the command value is negative (greater than 127), then the routine drops into a dummy loop that causes a delay. A shifted space has ASCII value 160 so it causes a delay. Otherwise control shifts to line 850, and a search is initiated to find the command in the index (lines 850-940). If the command is not found, it is ignored, and the program branches to get the next command from the stack.

When a command to change a feature is found, lines 950-990 transfer the screen pointer for the relevant feature to cells in the floating point accumulator (FACC serves as free zero-page memory for this routine). Then lines 1000-1080 set up a short subroutine in FACC to fetch bytes from the stored data. The screen pointer and the pointer to the stored data both are obtained from the Index entry for the command being implemented.

Lines 1110-1150 get a byte from the stored data and set indexes for current use. The data byte is tested in lines 1160-1210 to see if it is an ordinary datum or a special subcommand.

If the byte is zero, it means that all data have been transferred, and control branches to get the next Face command. Values one, two, and three, are special subcommands used to reduce the amount of space needed for data.

Value one is a skip command.. The byte following is fetched to determine how many screen cells to skip, and then the screen pointer is adjusted to accomplish this (lines 1330-1400).

Value two is a duplicate command. The byte following the two is fetched to define the character to be duplicated. The byte after that is fetched to determine how many times the character is to be displayed. Then the character is put on the screen the required number of times (lines 1420-1580).

Value three causes a feature manipulation to be appended after the current one. The byte following the three is fetched and pushed on the stack. Thus it will be the next Face command to be implemented (lines 1250-1270).

If the data byte is not zero, one, two, or three, then it is a character byte, and it is transferred directly to the screen (lines 1300-1310). Processing of data bytes continues in a loop until a zero value is found.

The subroutine in lines 1630-2030 searches the BASIC variable list for the FACE$ string. When it is found, the pointer to the string and the string's length are saved for use in the main program. If FA$ is not found, the procedure aborts via the BASIC error routine.

The Index shows the ASCII value for each command character, the place on the screen where the relevant feature begins, and the point in memory (listing 4) where data for that command are stored.

. *Charts and Listings occupy the rest of p. 33, pp. 34-36, and all of p. 37 except for the following.*

Additional Readings

Paul Ekman and Wallace V. Friesen, *Unmasking the Face* (Prentice-Hall, 1975), and Carrol E. Izard, *The Face of Emotion* (Appleton-Century-Crofts, 1971).

David Heise is Professor of Sociology at Indiana University. He recently edited "Microcomputers in Social Research" (Sage Publications, 1981). His books include *Causal Analysis* and *Understanding Events*. He plans to use the face program in a longer program for simulating social interaction. He may be contacted at the Department of Sociology, Indiana University, Bloomington, IN 47405.
