Chapter 6:  

CASE STUDY 2:  
CHORD KEYBOARDS

Introduction

Chord, or multipress, keyboards are keyboards on which operators type by depressing more than one key simultaneously. An example of chording on conventional QWERTY keyboards is the simultaneous use of the "shift" key and a letter key in order to type upper case characters. What distinguishes the keyboards discussed in this chapter is that this chording activity is the norm rather than the exception.

One attribute of chord keyboards is that they have fewer keys than there are symbols in the "alphabet" that they are used to type. Chord sets that are used to replace conventional keyboards, for example, typically have between six and eight keys. There are three important consequences of this. First, chord keyboards are physically small. This gives them a renewed appeal in light of the move to small compact computers. Second, because of the low number of keys, some designs support one-handed typing. Third, their operation is often highly moded, with shift sequences to specify things like upper case, numerics, control characters, etc. This moded behaviour (which means that the same chord has different interpretations depending on the context in which it is typed) can lead to increased learning time and operational error.¹

Using chords has two important effects on usage. First, if you can type at all, you can touch type. On the other hand, you cannot hunt-and-peck for chords that you don't know, or have forgotten. (As we shall see under the discussion of "Self Revelation", below, there is at least one design that addresses this issue.)

Chord Keyboards have two other important attributes. First, they afford one-handed typing. This is a benefit to disabled users (Kirschenbaum, Friedman, & Melnik, 1986), as well as those who want to occupy the other hand in some other task, such as pointing. Second, unlike stylus driven input and conventional keyboards, they are one of the only text entry methods that can be undertaken while mobile or in the presence of vibration, such as when walking, or in a bumpy train. This is another reason why their study is relevant to those interested in compact portable computers.

¹ The modes are partially a consequence of the low number of keys. With six keys, for example, there are only 64 (26) symbols that can be typed. This is less than the ASCII character set.
Chord keyboards have been studied for a number of years. The first reported example of a chord keyboard dates from 1942 (Conrad and Longman, 1965). An extensive review of the literature is given by Martin (1980), and a shorter version under her married name, Noyes (1983b). Seibel (1972) gives a good survey of relevant human-factors studies. These investigate issues such as the effect of number of buttons, number of chords, number of hands, and the relative difficulty of different combinations. None investigate the use of chord keysets used in combination with other devices.

Engelbart and English (1968) present a report on the early use of chord keyboards in human-computer interaction. They developed the piano-like keyboard shown in Figure 1. Their device illustrates the ability to support one-handed typing.

They viewed interaction as consisting of two classes of task: text entry and spatial pointing/selecting tasks. With a standard keyboard and mouse, this would mean that one hand would have to move back and forth between the keyboard and the pointing device. To overcome the inefficiency of this back and forth motion, they designed their chord keyboard. Using it, *text was entered with a combination of the chord keyboard and the mouse buttons* (Bill: check: *did he use mouse buttons or chord kbd alone*?), while the spatial pointing/selecting tasks were performed using a mouse\(^1\), (which Engelbart invented as part of this project). Since each hand would remain in *home position*, time spent moving the hand between the keyboard and the mouse would be eliminated (*homing time* as described in Card, Moran & Newell, 1980). Engelbart was able to achieve typing speeds of 35 words per minute with his right hand, and 25 words per minute with his left hand. It took him about 10 hours to reach 10 words per minute (Noyes, 1983)\(^2\).

---

\(^1\) It was this study that first introduced the mouse.

\(^2\) Note that there are other approaches to one-handed typing, such as the Maltron keyboard illustrated in Chapter 2.
Another early two-handed system that used one hand on a chording keyboard and the other on a pointing device was developed in between 1967 and 1969 at the National Research Council of Canada. To study human-computer interaction, an interactive system for music composition and a system for animation were developed (Pulfer, 1971; Wein, 1990). With the music system, for example, the right hand was used with two thumb-wheels to determine where in pitch and time a note was to be entered, and a chording keyboard (Figure 2) was used by the left hand to enter the note duration.

Another approach to pointing and keying information is to use the body of the pointing device itself as a chording keyboard. Some mice and tablet pucks are designed to facilitate this kind of interaction. Figure 3 shows one such example.

Chord keyboards and mice were widely available at Xerox’s Palo Alto Research Center through the 1970s. The keyboards were based on Engelbart and English’s “piano-like” design. They did not, however, achieve great popularity¹. However, that chording keyboards never caught on is not proof that they cannot provide powerful solutions to some user interface problems. The system for entering music discussed above (Pulfer, 1971) is one example. Furthermore, as Seibel (1972) points out, the fastest rates of keyed input have been achieved using chording keyboards.

¹ Dan Sweinhart of PARC recently made the following comments about the demise of the chord keyboard at PARC:

I always liked the chord keyboard - got pretty good at it. But there are some disadvantages, which I believe led to their abandonment at PARC/Xerox:

1) You have to learn how to use them. Teaching a new form of typing before a system could be used effectively was considered too large a start-up transient for most customers to learn. It was a marketing, and perhaps a human factors issue.

2) The chordset was originally used, along with the mouse, both to issue commands (such as D-W for “delete word”) and to enter small amounts of text. The editors in place at the time switched from mode to mode depending on the commands that had been issued, so that commands and text could be distinguished. When we switched over to the modeless, direct-manipulation style, the chordset was typically used simply for encoding various editing operations. The QWERTY keyboard had to be used for text. Users were forced to revert to the current style of switching back and forth from keyboard to mouse/keyset, and the value of the chordset faded away - function keys are easier to provide, and just as convenient.
SOME KEY ISSUES

With chord keyboards, speed of operation, proneness to error, and speed of learning are affected by a number of parameters:

- Physical Design: their number of buttons, their layout, and action.
- Size of Alphabet: how many symbols must be remembered
- Semantic level of symbols in alphabet: does a symbol represent an alphabetic character, a word, sentence ...
- Encoding scheme: the pattern assigned to each member of the alphabet
- Self Revelation
- Handedness: One handed or two handed (physically and cognitively)? If one handed, is it for the dominant or non-dominant hand, or can it be used equally well by either hand?
- Interference: cross-interference between chording and some other task

This list is not exhaustive, nor are its points mutually exclusive. They do, however, provide the basis for focusing our discussion.

PHYSICAL DESIGN

The symmetry of Engelbart and English’s keyboard permitted it to be physically used by either hand.

In contrast, some devices are designed to fit the specific shape of the left or right hand. One example is shown in Figure 4. Designing the physical ergonomics to the physiology of the hand can improve performance and comfort. (See Eilam, 1989, for example.) However, this same tailoring of the form also means that, in contrast to the piano-like keyboard shown in Error! Reference source not found., the device can be only used by the hand (left or right) for which it was designed.
This lack of interchangability can be a real issue. Consider the keyboard shown in Figure 5 the Microwriter.\(^1\) This is a portable word processor that uses a chord keyboard for text entry using the right hand. The device has an RS-232 interface that permits it to be used as the keyboard for a general purpose computer. In this case, it is quite conceivable that one may want to use a mouse in the right hand for pointing, and enter text on the chord keyboard using the left hand. This is the way that Engelbart used his chord keyboard (Error! Reference source not found.). With the Microwriter this is not possible. The same aspects of its design that makes it well suited for the right hand prevents its use by the left.

Another example of an asymmetrical design is the "Writehander" (Owen, 1978; NewO, 1978), shown in Figure 6. This device consists of four buttons mounted on a hemisphere so as to lie under the fingers. Eight other buttons lie within range of the thumb. The hemispherical shape of the device is worthy of note in that it takes advantage of the hand's ability to squeeze using a "power grip," much like a baseball. (The DePraz mouse\(^2\) shown in Figure 3 also uses this form, although its three buttons are mounted symmetrically, so that it can - in principle - be used by either hand.)

---

2 Available from Logitech, 165 University Ave, Palo Alto, CA, 94301
The reason that the Writehander and Microwriter can only be used by one hand is the positioning of the keys operated by the thumb. To achieve a keyboard that was not symmetrical, but which could still be used by either hand, Rochester, Bequaert and Sharp (1978) developed a chord keyboard on which the position of the thumb keys was adjustable. This is shown in Figure 7.

Figure 7: Keyboard Adjustable to Either Hand (from Rochester et al., 1978)

The number of keys on the keyboard is an important consideration. If there is only one key per finger, and the layout is appropriate, the hand can always remain on "home-row" (since it is the only row). Thus, if you can use them at all, you can touch-type. However, resting the fingers on the keys makes the problem of button quality (avoiding false key depressions), and their placement, critical. If you can rest your fingers on the keys without false depressions, then there is the danger that the key action will need too much force. On the other hand, if light action is desired, there is the danger of unintended depressions.

Key roll-over also presents a problem. What, for example, is the dividing line between two rapid sequential depressions and a sloppy chord? Among other considerations, this parameter may need to vary depending on the expertise of the operator. With the Writehander, the problem is side-stepped by having the chord transmitted upon depressing one of the thumb keys. The problem with this is that every chord must involve the thumb. Hence your ability to experiment with encodings is quite restricted.
The Accukey keyboard from Vatell Corp. (Kroemer, Fathallah & Langley, 1988), shown in Figure 8 is unique in that it uses three-state keys. The design utilizes a two-handed eight button keyboard. Each chord is a two-button combination. However, instead of the two up/down states of conventional keys, the keys move forward and backwards from a neutral middle position. Using this approach, consistent fingerings are used for a given character, for example, and the direction of motion of the keys determines which mode (function, shift, alt or control) is used. McMulkin (1992) presents a study of the learning curve of five users of this keyboard over 60 hours of use, using a limited 18 character vocabulary.

Figure 8: The Accukey Keyboard
This two-handed keyboard uses 3-state keys that move forward and backward from a neutral central position. (Photos: Vatell Corp.)

The Accukey keyboard from Vatell Corp. (Kroemer, Fathallah & Langley, 1988), shown in Figure 8 is unique in that it uses three-state keys. The design utilizes a two-handed eight button keyboard. Each chord is a two-button combination. However, instead of the two up/down states of conventional keys, the keys move forward and backwards from a neutral middle position. Using this approach, consistent fingerings are used for a given character, for example, and the direction of motion of the keys determines which mode (function, shift, alt or control) is used. McMulkin (1992) presents a study of the learning curve of five users of this keyboard over 60 hours of use, using a limited 18 character vocabulary.

Figure 9: Stenograph Keyboard
This chord keyboard is used in North America for court reporting. Speeds of over 200 words per minute can be achieved, but it takes about three or more years to achieve this level of performance.

Alternative designs for different classes of tasks should not be forgotten. Seibel (1962) developed a data entry station, "Rapid Type", that demonstrated the use of a modified QWERTY keyboard to enable chording input. By adding two extra shift keys, 150 common words became available by chording. His data suggest that keying rates can improve by up to 150% by using the technique.
A version of this technique, called RapidRiter, was introduced commercially but is no longer in production. The system was also based on a conventional keyboard, so users could type normally using their existing skills. The speed improvements came by permitting users to define abbreviations for frequently typed words or phrases. For example, holding down the "s" and "y" keys together could be an abbreviation for "sincerely yours," or "b" and "w" could be an abbreviation for "best wishes."

There are three main differences between this approach and most of the other designs discussed in this chapter:

The technique is not strictly chording. Chords supplement what one already knows (assuming one is familiar with the QWERTY keyboard).
- It builds upon the existing installed base of keyboards and skilled operators.
- The chord abbreviations are defined by the user. Hence, they are introduced gradually, and the issue of learning some predefined chord set is avoided. Of course if they are personalized, only that user can take advantage of the accelerators.

It is unclear why this product failed in the marketplace.

Finally, chording keyboards are commonly used to transcribe verbatim transcripts of speech. This requires transcription speeds of 180 words per minute, or greater, which is about two to three times typical typing speed. In North America, a Stenograph machine, such as that shown in Figure 9 is used for this, while in the United Kingdom, a device called the Palantype, (Figure 10) is employed. The high data entry speeds accomplished with such devices is commonly cited as evidence of the high bandwidth possible with chord keyboards. However, it is important to realize that in order to achieve such speeds, the operators must often train for three to six years. Furthermore, what is being keyed in during the transcription process is a phonetic script which must be transcribed into normal running text. Until recently, this transcription had to be done manually. Nowadays, this can be accomplished using a portable computer. Dowton and Brooks (1984) and Dye, Newell and Arnott (1984) are examples of early efforts to use achieve such automated transcription.

1 Quixote Corp., East Wacker Ave., Chicago, Illinois, USA 60601. tel: (312)-467-6755
SIZE OF ALPHABET

The larger the alphabet encoded on a keyset, the more difficult it is to use. A cognitive problem with such keysets is that you cannot hunt-and-peck. You must memorize the encodings, and this results in a longer training period, errors for infrequent chords, and re-learning problems for casual users. With the Microwriter, for example, it requires only about two to four hours to learn the 26 letters of the alphabet and the 10 digits. However, remembering all of the special symbols and punctuation is sufficiently difficult to discourage use of the device.

SEMANTIC LEVEL OF SYMBOLS IN ALPHABET

One reason that people have been attracted to chord keyboards is the prospect of improving the bandwidth of data entry by a human operator. But note that the rate at which an expert can "type" symbols on a chording keyboard is considerably slower than on a conventional typewriter. Devoe (1967) cites top rates of about 125 versus 800 strokes per minute respectively. Consequently, to increase bandwidth with chording, more information must be transmitted per stroke. We see this in Seibel's "Rapid-Type" which improved effective typing speed because common words were able to be abbreviated as a single chorded symbol.

Similar information "packing" can be used in input techniques other than chording keyboards. Programmable function keys are one example. In this case, the issue is the trade-off between number of keys and the number of symbols (words or characters). With a non-chording keyboard, $n$ keys gives us access to $n$ symbols, compared to $2^n$ using a chording version. The question of "semantic load" is also very relevant in designing input systems for people with motor disabilities. Here, due to the relatively high overhead of each action, the issue is to get as much information out of each one. Demasco and McCoy (1992), for example, give a good discussion of word-based virtual keyboards and sentence compansion as techniques to obtain maximum bandwidth per user action. In reading this work, it is important to realize that these techniques have potential application for the wider computer user population, and warrant investigation. Finally, we the topic of semantic load will reappear when we discuss marking interfaces in a later chapter. Here the issue is whether the mark being recognized represents a character, word or entire command (sentence).

SELF REVELATION

Self revelation is the property of user interfaces to reveal to the user how they are to be used, or what the current options are. For example, most menu systems are fairly self revealing, in that each menu item makes explicit one of the user’s options. On the other hand, command-line interfaces such as the Unix Shell, or MS-DOS, reveal little, if anything, to the user about what they can do next.

At the device level, the QWERTY keyboard is fairly self-revealing, since the labels on the top of each of the keycaps indicate what will happen if that key is pushed. However, even here there are problems, as illustrated by special characters that do not appear on a keycap. For example, most keyboards are not self-revealing when it comes to entering the "≠" character.
Since there is not a simple keycap-character mapping with chord keyboards, it is difficult to reveal to the novice how to enter specific characters, short of going back to the documentation. Hence the claim that in order to type on most chord keyboards, one must be able to touch type on them.

Whereas this is generally Science for the case, there are exceptions. Figure 11 illustrates a technique developed by the Australian Institute of Marine Science which addresses this problem for certain cases. What they do is display a menu on the screen which illustrates the effect of different chords. In the example, each option corresponds to one or more fingers (thumb at the left) on a 5 button hand-held controller. By pressing the thumb and first finger, for example, they can activate "Shell" as their next entry, similarly "Clear" would be all but the thumb, and "Mud" just the thumb.

The value of the technique is that it provides prompts for the various chord combinations. The weakness, however, is that it limits the chords that can be used, since only chords of contiguous keys can be conveniently labeled using this approach.

Figure 12: Encoding Characters and Words. (Rochester et al., 1978)

Figure 12 gives an example of how both characters and words are encoded on the keyboard developed by Rochester et al., 1978).
ENCODING SCHEME AND HANDEDNESS

The encoding used is critical in minimizing the problems of learning, retention, and operation. For one-handed keysets, it also has an effect on handedness.

Not all people will use a one-handed keyboard in the same hand. The obvious example is with left-handed and right-handed people. Even with a single individual, however, the device may need to be operated in either hand. When just entering text (with a Microwriter, for example), the major hand is typically used. However, when entering text while also using a mouse (as in the Engelbart and English setup), the text is often entered with the minor hand and the mouse manipulated with the major hand.

We have already seen how the design can facilitate, or impede, the ability to physically use the device in either hand. With the Engelbart and English keyset, transfer is easy because of the symmetrical piano-like design. However, with the Writehander and Microwriter a separate physical device - designed especially for the other hand - must be used.

Hand-to-hand transfer presents even greater problems. At issue is how the codes memorized on one hand transfer to the other. Will the encoding on one hand be the "mirror image" of the other, or will spatial congruence be maintained? If the keyboard has a vertical orientation, the two will be the same, and the issue disappears. However, this is not the case with any of the one-handed keyboards discussed.

We can gain some insights about the hand-to-hand mapping by looking at errors in conventional two-handed typing. Figure 14 presents data from Munhall and Ostry (1983). If you compare the spatial congruence and the mirror-image pairs, you see that mirror-image substitutions occur much more frequently (in some cases, as much as 10 times more often than spatial congruence substitutions). This high frequency of mirror image substitution errors suggests that this mapping will be most likely when transferring the operation of a one-handed keyboard from one hand to the other.

**Figure 13: Encoding Scheme for Microwriter. (Microwriter Ltd.)**

**Figure 14: Errors in 2 Handed Typing (from Munhall and Ostry, 1983).**
In contrast, a study of Gopher and Koenig (1983) suggests that the codes for one-handed chording keyboards will most naturally transfer by spatial congruence. If we examine the Microwriter, however, we will see how the question is largely a result of the encoding scheme used. In learning the codes for the Microwriter, mnemonic aids are introduced. In fact, at least three different types of mnemonic are used:

- **kinesthetic based**: for example, the most agile (index) finger is used for the most common character, 'E' (similarly, thumb is used for second most common character, ').
- **word association**: for example, "S" is typed using the "signet" ring finger.
- **spatial mnemonics**: for example, 'L' is typed by pushing three keys spatially corresponding to the three vertices of the graph of the character. The encoding for 'J' uses the same scheme.

The encoding scheme for the full alphabet is shown in Figure 13. We will not deal here with the questionable practice of mixing mnemonic types (if you can't remember a character, you must first remember how it was encoded). What is important to note is that (in contrast to the findings of Gopher and Koenig, 1983) the first two encoding schemes will always transfer from hand-to-hand in a mirror image.

Conversely, the third scheme will nearly always maintain spatial congruence in transfer. (Otherwise, the mnemonic for 'J' would be shaped like an 'L' when we transfer hands.) Thus, we see that we can choose our encoding scheme and training to encourage one form of transfer or the other. Furthermore, if we mix the two types, as did the Microwriter, we severely impede the operator's ability to transfer the skill from hand-to-hand.

The implications of these issues go beyond the obvious. With longer alphabets, it could be argued that spatial congruence should be preferred (Gopher & Koenig, 1983). However, many mice can and should be considered chording keysets. But they generally only have 2 or 3 buttons and few commands. In this case, there is a strong argument to use mirror image encoding and have the most common (select) function triggered by the strongest (index) finger.

These are subtle but important issues that must be dealt with. Consider, for example, a 3-button mouse used with mirror image encoding, as suggested. Typically, users (left and right handed) use the button lying under the index finger for selection. Due to the asymmetry of the hand, however, these would be the right and left buttons, respectively. To respond correctly, the system must interpret input according to which hand the mouse is in, and references in the documentation should refer to the finger rather than the button of the device.

But all of this can get us into trouble. Let us accept that a fundamental characteristic of good design is self consistency. We just argued for a mouse implementation based on mirror-image hand-to-hand transfer. But what happens when this mouse is used in combination with a chording keyset? If the keyset transfers encoding by spatial congruence, do we have an inconsistency that will affect the quality of the user interface? Yes, if at different times both devices must be used by both hands for a single user. No, if only one of the two devices transfers from hand-to-hand. Both cases can occur. Note that the issue of mirror image argued for in the case of the mouse was as much to improve the ergonomics for left-handed users as to enable one user to operate the mouse in either hand. With the chord keyset, the issue was hand-to-hand transfer for a single operator.

**INTERFERENCE**

All other things being equal, it should be no surprise that performance using a chord keyboard can be affected by other tasks. One such case is in two handed input where one hand is typing on the chord keyboard, and the other performing spatial tasks with a mouse. In this type of dual task, coordination of the two hands can be difficult, and require significant training. Therefore, task assignment and difficulty must be carefully considered and tested.
Another type of interference occurs when the typing and spatial tasks are performed by the same hand. This is seen, for example, when chording with the mouse buttons. Here, trying to type while simultaneously positioning the mouse will generally degrade the performance of both tasks. Typing even when the mouse is stationary may cause problems. Generally, the buttons of mice and tablet pucks are mounted so that their action is as close to vertical as possible. This is to keep the direction of button motion orthogonal to the plane of motion in positioning. However, buttons so positioned are generally not in the most comfortable location for chording. Hence, the DePraz mouse, which is optimized for chording, is held much like the Writehander, so that the finger tips rest comfortably over the keys. The penalty for this, however, is that the direction of force in button pushes is parallel to the plane of motion in positioning. Furthermore, in this position the mouse is held in a "power grip" (much the way you hold a ball). As a result, mouse positioning must be carried out using the larger muscle groups of the arm, rather than the wrist and fingers, which offer finer control.

One consequence of this discussion is a realization that a factor in choice of mouse or tablet puck should be whether the keys are to be used for chording or not, and what type of positioning is demanded.

CONCLUSIONS

Based on the literature and personal experience, I believe that chording keyboards have an important role to play in human-computer interaction. As Norman and Fisher (1982) point out, major improvements in methods for keyed input will only be achieved through a radical change from current practice. This is true for both novices and experts. The use of chord keyboards as an alternative means for keyed input is still under developed. This is, I feel, due to the range and complexity of the issues affecting their performance. To change this situation, research must investigate appropriate applications as much as technical issues. In my mind the issue is not if chord keyboards can be effective, but where and how?

TO DO STILL:


Add figure 14