Abstract

A new hand- or finger-mounted data input device is presented, using traditional touch-typing skills as method of (alphabetical) character input thus providing an ultra-portable solution for “quiet” data input into portable computer systems. The presented keyboard independent touch-typing device (KITTY) offers high data input rates and a virtually zero learning curve for new users.

1 Motivation

With recent advances in portable computing and in particular the design of pocket PCs [13] and eye-glass displays [14], the development of new augmented reality (AR) environments has been made possible. However, one of the current shortcomings in these AR systems is the lack of intuitive input devices that provide users with complete control over their workspace. In particular in environments where voice input is undesirable or infeasible, touch-typing capabilities have to be available to allow for intuitive access to possibly complex data.

2 Introduction

The size, layout and weight of traditional keyboards make them poor candidates for use in mobile AR applications. In particular, if integration with mobile computing devices, such as personal digital assistants (PDAs) or pocket personal computers (PocketPCs) is desired, a new generation of natural handheld input devices is required.

Today’s mobile computing devices frequently include a physical or digital miniature keyboard, that may be integrated with the device, worn on the user’s wrist or simply be part of the display area. Unfortunately, these keyboards generally require the use of both hands, one holding the device and the other for finger-based or stylus-based data input. The selection of individual keys on a hand held display-based keyboard, for example, is a rather inefficient process and requires the user to look at the keyboard devise. Handwriting recognition can increase input rates after extensive training but is still very limiting. This performance gap is problematic if the portable computing devices are to be operated at the level suitable for today’s computational platforms.

Other devices for data input into portable computing units, include chording devices, such as a hand-mounted keypads that require the user to press different key combinations to generate the characters found on a standard keyboard. Further alternatives include microphones in combination with voice recognition software. The former input devices typically do not meet the data input rates desired for full-scale interaction and typically require the user to learn a chording/coding language particular to the specific device. On the other hand, privacy concerns or ambient noise frequently limit the use of voice recognition devices. Possible scenarios include input of confidential or sensitive data via voice recognition in conference settings, meetings, or crowded public places, such as airports.

3 Related Work

With a demand for decreased keyboard size and increased portability, while maintaining the traditional keyboard layout, new device concepts are required. A first attempt was
the development of foldable keyboards that allowed users to carry familiar input devices with reduced physical footprint. While these devices are certainly more compact than traditional full-size keyboards during transport, a folding keyboard is still relatively large in the unfolded state in which its surface requirements match that of a traditional full-size device.

Other devices have tried to reduce the number of keys rather than reducing the overall size of the keys in order to achieve size reductions. For example, one-handed keyboards use only half of a traditional keyboard by mapping two characters on each key. This requires the addition of a switch (keyboard identification) key, enabling the user to toggle between two sets of characters, each representing one-side of a full-size keyboard. Even though “half-keyboards” may have a very small footprint, they commonly still require a support surface.

The “body-wearable keyboard” and similar the so-called “Gesture-Pad”, which is under development by Sony Inc., place “keyboard-keys” on the users clothing or a skeletal structure worn on the users body. Even though this increases portability, it greatly restricts the users arm-movement and flexibility for data input.

Many of the existing keyboard- or keypad-less data input devices for portable computing applications, use finger or hand mounted sensors in order to generate the signals that would normally be generated by pressing a keyboard key. The “finger-mounted computer input device” [11], the “fingerless glove for interacting with a data processing system” [15] and the “Virtual Keyboard” by Senseboard Technologies [12] use pressure sensors on the users fingers or sensors (mounted on gloves) measuring flexure of the fingers in order to determine data input via finger positions with respect to a physical or virtual reference surface. Generally, these devices assume that the reference surface functions as virtual touch-typing environment. As for folding keyboards, the need of a physical support surface greatly limits the mobile computing capabilities of these devices.

In response to this limitation, hand- or finger-mounted input devices have been proposed, such as the "Chording Glove" [16] and "Twiddler" [1]. The Chording Glove has contacts at various positions on the fingers and thumbs and a combination of contacts (chord) is pressed simultaneously to generate a specific keyboard event. Although chording devices remove the need for a support surface, their use requires substantial training as the user has to learn a new coding language.

Vaughan R. Pratt describes a device-independent digital sign language (Thumbcode), using keys on the users fingers that are activated by the thumbs of the same hand [10]. In order to generate different input signals with the same key, the user has to pair neighboring fingers according to a prescribed coding language. As with chording, thumbcoding requires a user to essentially learn a new coding language in order to enter data into the computing device.

The functionality of the so-called Lightglove by Howard [4, 3], is based on the generation of a “light-matrix” below the users hands. Penetration and location of penetration of this matrix by the users fingertips is detected and electronically mapped to an overlay assigning each position a character according to the layout of a traditional keyboard. This operation of the device involves simultaneous scanning of the matrix with various light emitting diodes or lasers, evaluation of the scattered electromagnetic waveforms by various light sensitive detectors and signal filtering in a bandpass filter in order to reject non-correlated ambient signals. Generation of the optical matrix and detection of finger position in this matrix is fairly complicated. Furthermore, the system has to be calibrated (possibly in real-time) for specific ambient conditions, such as strong ambient illumination or electromagnetic interference. The lack of tactile feedback (providing confirmation of data input as is the case for touch-typing on a keyboard) may also result in lower data input rates. The use of an optical reflection matrix to detect signal input by predicting finger position with respect to the optical matrix below the user’s hands does not allow for free finger motion without the risk of data input. Instead, the user has to hold the hands and fingers in a more-or-less stretched out position in order to omit interference with the optical matrix. The involuntary generation of signals is a common problem with these devices and has to be addressed. Finally, many of the devices described in this section are large, counter-intuitive, and too cumbersome for fast data input. Most users when confronted with having to learn an entirely new coding language in order to enter information will likely decide against using such a device. This is particularly true for good VR or AR applications that should require only very limited user training.

For VR/AR and mobile computing applications, intuitive, flexible, and efficient, keyboardless input devices are urgently needed. KITTY, a device prototype that addresses these objectives is presented in this paper (Figure 2).

4 KITTY Interface

The presented KITTY interface overcomes the limitations described in the previous sections by providing intuitive access to common touch-typing skills. To accomplish this, the device provides multiple contacts on each finger that are only activated when the appropriate finger and thumb contacts are combined. The layout of these contacts follows the traditional QWERTY keyboard layout, allowing anyone with previous touch-typing experience to immediately use the device. A significant advantage of this device is that arbitrary finger motion is allowed without data input, as long as finger contacts and thumb contacts on a single hand do not meet. In other words, KITTY allows the user complete freedom for
4.1 Finger-Thumb Touch-Typing

Two different key-maps were proposed for the finger-mounted KITTY interface. Section 4.2 discusses the alphabetic key-mapping as employed in the wired prototype shown in Figure 8. The mapping of alphabetic keys is described in the framework of a comprehensive mapping procedure for the entire set of keys/commands found on a standard keyboard. Note that, the wired prototype shown in the photograph employs a commercially available keyboard by GrandTeck™ and provides alphabetic data-input functionality as well as punctuation marks and a limited number of essential commands such as “Delete”, “Space”, “Return”, and “Shift”. Section 4.3 describes a slightly modified key-map procedure for alphanumeric character input. This mapping is proposed for the new wireless KITTY interface currently in development.

Figure 3 illustrates a standard QWERTY keyboard, which was used as the reference for the design of the KITTY keyboard mapping. While the QWERTY layout was chosen for the initial implementation other key mappings such as Dvorak or the half-keyboard can be easily supported.

Using the standard QWERTY keyboard (Figure 3) via touch-typing, a user’s fingers are normally in a home position that includes placing the left pinky on the A-key, the ring finger on the S-key, the middle finger on the D-key and the left index finger on the F-key. The home position for the right hand includes placement of the right index finger on the J-key, the right middle finger on the K-key, the index finger on the L-key and the right pinky on the :key. This row (ASDF ...), which is annotated in Figure 3 with a solid line, may be referred to as the base row. The row above the base row (QWERTY ...) is annotated with a long dashed line and is referred to as the top row and the row below the base row (ZXCV ...) is annotated with a short dash line and referred to as the bottom row. Figure 3 also illustrates the enumeration of left-hand fingers (1-4), right-hand fingers (5-8) and thumbs with thumb contacts (A-A0), (B-B0), (C-C0), (D-D0), (E-E0), (F-F0) employed for the key-map of the wired prototype as discussed in Section 4.2.

4.2 Key-Map Wired Prototype

Figure 4 illustrates the location of electrical contacts on the user’s hands as employed within the present wired prototype and following the key layout shown in Figure 3. Both finger- and thumb-contacts are employed. An electric circuit is closed and a signal is generated upon closing of one finger-contact with one thumb-contact as discussed below in detail.

As shown in Figure 4 by dashed lines, finger contacts 1-8 are located on the palm-side of the user’s hand near the tops of the fingers or the fingertips. There is one finger contact on the pinky, ring, middle and index finger of each hand (1-8). Also, there are six contacts A-F, A0-F0 on each thumb; three contacts on each inner thumb and three contacts on each outer thumb. As described in more detail below, signals for letters A-Z are generated by contacting one of the
finger contacts 1-8 with one of the six thumb contacts A-F, A0-F0 on the same hand.

As shown in Figure 4, there are supplemental finger contacts located along the sides of the user’s fingers nearest to the user’s thumbs. Any given finger may have three supplemental contacts located proximate to the first, second and third segment of the user’s fingers. These supplemental contacts are used to generate signals representing special characters. For example, the supplemental contacts located near the base and middle of the ring and pinky fingers of the right hand are used to generate the same signals that would be generated by the arrow (left, right, up and down) keys. A signal is generated by contacting a particular supplemental contact with the thumb contact A or D located on the inside tip of the thumb of the same hand. In other words, “Thumb-coding” is used for these special characters due to the infrequent use of these characters.

As illustrated in Figure 5, additional contacts (denoted by numbers 1-10) may be located on the fingernails and/or neighboring finger segments of each hand for input of numbers 0-9 and (upon using ‘Shift’) several ‘special’ characters. Data input is achieved by closing contacts 1–5 with thumb contact A and contacts 6-10 with thumb contact B, respectively.

Table 2(a) through 2(f) illustrate finger-thumb contact combinations employed for the wired prototype in order to generate the signals that would be generated by pressing the keys on a standard QWERTY keyboard. The middle thumb contacts B, E are used for characters in the base row. The top thumb contacts A, D are used for characters in the top row and the bottom thumb contacts C, F are used for characters in the bottom row. For example, on a QWERTY keyboard the character “a” is the character that would be typed using the index finger of the left hand in the home (base row) position. With the present method, the signal representing the character “a” is generated when the finger contact 1 on the pinky finger of the left hand (see Figure 4) is closed with the thumb contact B (representing the base row). Since the left pinky finger is used to press the “q” character located on the upper row of a keyboard, the signal representing the character “q” is generated by closing the contact between the pinky finger contact 1 and the top thumb contact A (see Figure 4). As can be seen in Figure 3 and Table 2(b), the combination of finger contact 1 and thumb contact A is used to generate the signal representing the character “q”. Since the index finger is used to reach two characters on a given row, an additional thumb contact for each row is located on the back or outside of the user’s thumbs. See contacts A0-F0 in Figure 4. For example, to generate the signal representing the character “f”, the finger-thumb-contact combination 4-B is used. The contact combination used to generate the character “g” is 4-B0 which implies contacting the left index finger 4 with the center contact on the outside of the user’s left thumb B0 (see Figure 4). Note that, due to ergonomic reasons, finger-thumb contact 1-C is not used to generate “z” but rather 8-E. Accordingly, “;” and “.” are generated by one of the supplemental contacts located on the sides of the fingers as shown in Figure 4.

The previously described method to generate numeric data input (using contacts 1-10 in Figure 5), avoids the need of placing four thumb contacts on the inside of each thumb in order to mimic a total of four rows on a standard keyboard. The latter is problematic considering the finite size of the electrical contacts and the limited area on the inside thumb which is easily accessible by the finger contacts on the palm-side tips of the fingers.

4.3 Key-Map Wireless Interface

Table 1 illustrates an alternate keyboard mapping based on four thumb contacts (T1-T4) and six finger contacts (LP, LR, LM, LI1-LI3) placed on the pinky, ring, middle and index fingers as shown in Figure 6. Contact T1 is placed on the thumb nail, T2 on the thumb pad, T3 on the center of the thumb pad and T4 on the bottom of the thumb pad (just below the knuckle). As for the prototype key-map described in Section 4.2, finger-contacts represent rows and thumb-contacts represent columns on a standard keyboard. Here, a new labelling is employed to allow the reader to easily distinguish
Table 1. The KITTY keyboard to hand mapping based on four contacts per thumb (T1-T4) and six additional finger contacts for the left (L) and right hand (R).

<table>
<thead>
<tr>
<th></th>
<th>LP</th>
<th>LR</th>
<th>LM</th>
<th>LI1</th>
<th>LI2</th>
<th>LI3</th>
<th>RI3</th>
<th>RI2</th>
<th>RI1</th>
<th>RM</th>
<th>RR</th>
<th>RP</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>Q</td>
<td>W</td>
<td>E</td>
<td>R</td>
<td>T</td>
<td></td>
<td>Y</td>
<td>U</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
</tr>
<tr>
<td>T3</td>
<td>A</td>
<td>S</td>
<td>D</td>
<td>F</td>
<td>G</td>
<td>Shift</td>
<td>Shift</td>
<td>H</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>”,”</td>
</tr>
<tr>
<td>T4</td>
<td>Z</td>
<td>X</td>
<td>C</td>
<td>V</td>
<td>B</td>
<td>Space</td>
<td>Space</td>
<td>N</td>
<td>M</td>
<td>&lt;,</td>
<td>&gt;,</td>
<td>? /</td>
</tr>
</tbody>
</table>

Figure 6. Finger contacts for wireless KITTY implementation.

Figure 7. Keyboard to glove mapping of wireless KITTY implementation.

According to the alphanumeric mapping summarized in Table 1 and illustrated in Figure 7 each finger contact is linked to the column of characters it would press on a keyboard using touch-typing. For example, the middle finger of the left hand, LM, is linked to "e", "d", and "c", the same keys this finger would normally press on a keyboard using touch-typing. The three thumb contacts match the three letter rows on a normal keyboard. The contact at the tip of the thumb, T2, corresponds to the top row of the keyboard, T3 to the base row, and T4 to the bottom row. Using the left middle finger again as an example, LM would yield "e" when touched to T2, "d" when touched to T3, and "c" when touched to T4.

In this modified key-map, each index finger has two more contacts on the side of the finger, LI2, LI3, RI2, and RI3, spaced above and below the top knuckle. The contact above the top knuckle, LI2 or RI2, corresponds to the second column of keys ("t", "g", and "b" for the left index and "y", "h", and "n" for the right index) that the index finger presses on the keyboard during touch-typing. The location of this contact follows the sequential feel of moving from finger to finger and column to column. LI3 and RI3 are used for the "Shift" and "Space" keys, which have no finger contacts that match the row-column arrangement. Keeping in mind, however, that the "Space" key sits below the "Shift" key, the third index contact is paired with T4 for "Shift" and T3 for "Space". The T2 contact is reserved for a missing number key as discussed below.

Numeric input is provided by simply placing another contact, T1, on the back of the thumb on the nail. The finger contacts are touched to T1 consecutively in keeping with the traditional keyboard assignments. This configuration for numbers works also well for non-touch-typists since it follows a traditional method of finger counting. Starting at the left pinky, LP touches T1 to generate the number 1, and then LR, LM, and LI1 follow for numbers 2 to 4. The number 5 is the missing number that uses the LT3 to T2 combination. Although an odd combination, it still follows the flow of the series as the tip of the thumb is used for number 5 like the finger pads for the other numbers. The right hand picks up number 6 in a mirror series to the left hand, with 6 to 0 moving through RT3, RT1, RM, RR, and RP in that order.

As with the key-map discussed in the previous section, a drawback to the described key-to-glove mapping is the bottom row key on the pinky. Touching LP or RP to T4 has an “awkward feel” to it. A momentary slight strain is felt in stretching the pinky towards the lower pad of the thumb. Like learning an awkward chord on a guitar, it is possible that the strain will lessen with prolonged use as the hand adjusts towards this unaccustomed position. One could argue that the keys "Z" and "? /" are used so infrequently that there is no concern regarding continuous straining of the tendons. If it is found, however, that the awkwardness and discomfort of the particular finger combination becomes an issue, an additional contact could be placed on the sides of the middle fingers to offer an alternative and a more comfortable combination.
5 Hardware Implementation

Two different implementation strategies were investigated supporting a wired and wireless configuration of the KITTY device.

5.1 Wired Prototype

Initially, the KITTY interface was implemented with minimal effort using the “Virtually Indestructible” foldable keyboard by GrandTec™ (see Figure 8). The mentioned keyboard was chosen due to the 100% correspondence between the circuit layout on the two membranes of the keyboard and the key-mapping described in Section 4.2, with finger-contacts representing keyboard columns and thumb-contacts representing keyboard rows. Wires were attached to the corresponding locations on the keyboard membrane circuitry using self-adhesive copper tape. Wirewrap was used to keep the design sleek. The wire ends were then collected and attached to two golf gloves according to the layout shown in Figure 4.

![Figure 8. First prototype system utilizing traditional roll-up keyboard as the keyboard encoder.](image)

Clearly, the wiring or printed circuitry on the two membranes of the “Virtually Indestructible Keyboard” by GrandTec perfectly matches (without modifications) the mapping proposed in Section 4.2. The GrandTec™ keyboard uses two membranes, membrane A onto which the wires for the finger contacts are attached and membrane B onto which the wires for the thumb contacts are attached. The contacts for the letters on membrane A are connected in a column format and the contacts on membrane B are connected in a row format.

5.2 Wireless Interface

Once a finger-thumb contact has been established according to the mapping schemes discussed earlier, various methods can be used to generate the signals. For example, wires from the contacts may be connected directly to a keyboard encoder. The keyboard encoder may be located on top of the user’s hand. For example, the keyboard encoder may be attached to a glove, such as the one shown in Figure 9. The signal that is generated can then be transmitted to the computing device via wire or wireless, e.g., infrared (IR) or the like. The computing device can then display the character represented by the signal.

**Keyboard Encoder** – A 8x12 keyboard encoder would suffice for the keys mapped according to Section 4.3. However, a 8x16 matrix is preferred; the larger matrix allows future functionality for additional keys such as macro keys. The smallest possible platform and low power consumption are other factors in deciding which keyboard encoder to use. A small platform is necessary to keep the wearable device unobtrusive and comfortable to wear for the user. Since power would be supplied by batteries, it is important that power consumption be minimized for longer battery life. For the above reasons, the Motion Encoder ME1001A keyboard encoder was selected for the wireless KITTY interface.

Thumbs contacts are wired to the row pins of the encoder, while the finger contacts connect to the column pins. The encoder transmits a single keystroke as a TTL serial signal at 2400 bps with no parity, 8 data bits and 2 stop bits. A hexadecimal number from 0x00 to 0x7f (decimal 0 to 127) indicates which key on the matrix was selected. This number is transmitted as part of a byte and occupies bit positions 0 to 6. The seventh bit indicates the key state of key down or up. The recommended 6MHz resonator is attached to the encoder. Other additional components that must normally be added to other encoders, such as pull up resistors, are internal to this encoder and helps keep the size of the device as small as possible.

**Wireless Signal Transmission** – The KITTY interface was created with portability and convenience in mind. Accordingly, it was our goal to add wireless signal transmission capabilities to a second generation KITTY interface providing data-input via gloves which are physically disconnected of the computing device itself. From the wide range of wireless technologies available, Bluetooth™ technology has been used. Bluetooth™ wireless technology is being used extensively in wireless and portable devices like cell phones and PDA’s, and has become a wireless standard. It provides plug and play type connectivity between any two or more Bluetooth™ devices. One of the goals behind the ongoing development of KITTY is to provide an input device that could work with PC’s, PDA’s, Pocket PC’s and any other device that requires alphanumerical input. Hence the use of Bluetooth™ wireless technology was appropriate for this application.

Within the new wireless KITTY data-entry system, the serial output of the encoder is fed into a Bluetooth™ RS232/UART DCE Terminal that is configured to communicate serially with the encoder. For this prototype the Wire-
A BlueWave RS232 PCB DCE Terminal is used that connects to a generic Bluetooth™ USB dongle/adapter that supports the Bluetooth™ serial port profile. This DCE terminal was chosen for its programmability, low power consumption and its size. The prototype has been developed for a Windows 9x and Windows NT platform. The adapter used in this case is the D-Link DBT-120 Bluetooth™ USB adapter. It does not require any external power supply, but receives all necessary power from the USB port. The DCE terminal requires a supply voltage that is between 3.3V to 9V DC, which is accomplished by using four AA Ni-MH rechargeable batteries that provide 1900mAh each and a total potential difference of 5.2V. The terminal’s average current consumption is 1.8mA when it is in the idle mode and 60.9mA while it is transferring data.

The DCE terminal measures about 40mm x 30mm x 5mm, and the antenna extends about an inch over that. The voltage is applied across pin 1 and pin 12 of the DCE terminal with pin 1 connected to the positive terminal and pin 12 connected to the negative terminal of the 5.2V battery pack. The serial data is connected to the TxD pin, which is pin 6 of the DCE terminal. Pins 3 and 4 are connected to each other in order to disable any hardware flow control. The serial data signal and the voltage supply are connected to the DCE terminal using a 0.05” (1.27mm) pitch ribbon cable.

Both the adapter and the DCE terminal conform to the Bluetooth™ v1.1 standard. When power is switched on for the DCE terminal it will become “discoverable” by other Bluetooth™ devices. Since it is a DCE terminal it can only be used as a slave module. The Bluetooth™ serial port profile creates a virtual COM port on the PC, which is used for connecting the two devices. The virtual COM port is similar to a physical COM port, but instead of having a wire for transferring the data, the data is transferred wirelessly. The serial data from the encoder is received by the terminal and is transmitted to the adapter. The virtual COM port is also configured to the same setting as the encoder and the terminal. Then a software driver decodes the data received and produces the appropriate keystroke. The information is transmitted at 2400MHz to 2483.5MHz frequency, as specified by the Bluetooth™ standard. The driver is responsible for configuring the virtual Com port, waiting for the data, decoding the data and producing the appropriate character using the keybd_event() system call.

5.3 Design Concepts

Various designs have been considered for implementing the described KITTY interface. The employed prototype design shown in Figure 1 and Figure 9 uses contacts attached to gloves (only one glove is shown in the figures carrying only contacts for alphabetic character input). The contacts are connected by wires to the encoder and (for the wireless implementation) transfer electronics. Preferably, the gloves are fairly thin and lightweight, thereby allowing the user to easily manipulate his fingers.

An alternative configuration also shown in Figure 9 has contacts mounted on clips or half-rings. The clips are attached to a skeletal structure. The keyboard encoder and transfer electronics (e.g., wireless transmission Electronics) are also mounted to the skeletal structure. Figure 9 shows an exemplary skeletal structure for the left hand. The skeletal structure rests against the user’s palms (or alternatively on the back of the user’s hand) and the contacts are positioned on the user’s fingers by placing the clips (on which the contacts are mounted) around the user’s fingers and thumbs. The clips can be attached to the skeletal structure in a way that allows adjustment of the positions of the clips in order to accommodate different finger and thumb lengths. The skeletal structure can also be adjustable in order to accommodate hands of varying sizes. The main purpose of the skeletal structure is to preserve the integrity of the overall system.

With recent advances in portable computing and in particular the design of pocket PCs and eye-glass displays, efforts aimed at the design of portable input devices and techniques have increased. While voice recognition is becoming
a strong contender for next generation input technology, privacy concerns and ambient noise problems are still a challenge and alternative approaches are required. The presented KITTY interface consists of a finger-mounted device mimicking finger movements similar to traditional touch-typing in order to provide alphanumerical data input. This low cost device guarantees a rapid learning curve for everyone experienced in touch-typing while offering significant flexibility. Two prototypes were built one using a PSII and one using a USB version of the GranTec keyboard. Priced at around $50 US for parts the KITTY device proved to be a very cost effective solution. The proof-of-concept implementation of the KITTY device was tested with wearable devices and desktop systems and demonstrated its effectiveness.

References