

# Paradiddle: An Exploration of a One-Handed Tap-Based Text Entry Technique

Seungyon “Claire” Lee

College of Computing, Georgia Institute of Technology  
TSRB, 85 5th street, NW, Atlanta GA 30332  
sylee@gatech.edu

Mark W. Newman, Kurt E. Partridge

Palo Alto Research Center  
3333 Coyote Hill Road, Palo Alto, CA 94304  
{mnewman,Kurt.Partridge}@parc.com

## ABSTRACT

In this paper, we present Paradiddle, a tap-based one-handed text-entry method. Paradiddle is designed to support mobile text entry where, for many applications, the total time to enter text (text-entry latency) may be more important than raw speed (text-entry bandwidth). Like many text-entry systems, Paradiddle uses quick taps to select letters, but Paradiddle distinguishes taps not primarily by position on the user interface surface, but rather by the finger that performed the tap. We test the Paradiddle technique by comparing an implementation that approximates its user experience with Twiddler. Although the results show that Twiddler is faster to learn, exploratory modifications show promise for improving Paradiddle’s performance.

## Author Keywords

One-handed text-entry, surface-based text-entry, arpeggiated finger taps, spontaneous text-entry, multitasking, chorded text entry, wearable computing

## ACM Classification Keywords

H5.2. Input devices and strategies, User-centered design, prototyping,

## INTRODUCTION

*It is 10:30 am at the airport. Julie is late for her flight at 11:00 am. Although she already has a shoulder bag with her, she decides to carry her suitcase since she does not have enough time to check it. At the self-service check-in counter, she happens to meet Don Smith. They haven’t seen each other since they graduated high school ten years ago. Because Julie needs to rush, she quickly asks his phone number, and then runs toward the security line, hoping to remember Don’s number long enough to enter it into her*

*laptop, cellphone, or other organizing device.*

Why can’t Julie immediately record Don’s number? Even though she only needs to enter 10 digits, she must also perform many other actions: parking her suitcase, opening her shoulder bag, searching her bag for the phone, retrieving the phone from the bag, turning the phone on, navigating the phone’s interface, entering Don’s name, turning the phone off, stowing the phone, returning her bag to her shoulder, and re-grasping the suitcase handle.

This situation is typical of many situations where a user needs to interact with information devices or services that may be embedded in the surrounding physical or virtual environment. The interaction needs to be quick, seamless and ready-at-hand. In situations like these, the text entry words-per-minute rate is not the dominant factor. Instead, what matters is the total amount of time spent interrupting a user’s current task. Viewed as a communications channel between the user and the device, in these situations it isn’t the **bandwidth**, but rather the **latency** that is crucial.

One way to minimize this human-device latency is by use of wearable computers. With head-mounted displays and handheld chording keyboards such as the Twiddler, one can quickly switch from a “real world” task to a computer interaction. Using only one eye and one hand, it is possible to multitask computer use with many everyday tasks, such as shopping, cooking, driving, or even nursing an infant.

However, a drawback of a hand-held wearable text-entry device is that to really minimize the latency, the user must maintain the alignment between his or her fingers and the device’s buttons. If the user must use the hand for another task, then switch back to text entry the user must reacquire the device and realign their fingers to it.

If it is possible instead to reduce the overhead surrounding text entry episodes, we can more seamlessly integrate text entry into the flow of other tasks. This paper presents a step toward this vision of instantaneous text entry: a one-handed, tap-based text-entry technique called the Paradiddle technique.

Using the Paradiddle technique, a user enters a character by tapping his or her fingers in a unique pattern. For example, thumb followed by ring-finger followed by the thumb again might enter a “d.”

Because it is the fingers, and not the tap positions, that determine the letter patterns, a keyboard or button-based device does not meet the requirements for the Paradiddle technique. Implementations that do meet the requirements are touch-sensitive surfaces that accurately track finger position, and wrist-worn or finger-worn devices that detect individual finger tapping. In addition, because buttons and other physical elements of the tapped-on surface are not used, users need not align their fingers as carefully with the device in order to begin typing. For a touch-sensitive surface implementation of the Paradiddle technique, the user need only acquire the device surface, which might be felt through a jeans' pocket or purse's fabric. For a wrist-worn device or finger-worn devices, the user need only acquire any tappable surface such as a wall or table.

Our solution builds on prior implementation breakthroughs [2,9] that enable buttonless tracking of finger taps. However, we wish to extend this work by developing a method that can be implemented on more ubiquitous input technologies such as resistive panels that can only detect one pressure point at a time. Realizing the vision of low-latency text entry will require additional efforts that are beyond the scope of the present work, such as designing user interface mechanisms to route the entered text to the correct device, service, or application, providing input feedback with low cognitive demands, and assessing the latency benefits.

This paper covers an important step toward this vision: a prototype implementation of the Paradiddle technique (which we call simply "Paradiddle"), and a comparison to Twiddler in terms of text-entry input bandwidth. The analysis shows that Twiddler performs better than Paradiddle, but that there are many opportunities for improving Paradiddle's design.

### RELATED WORK

Many other devices and mechanisms for mobile text entry have been proposed, most of which are button-based and not fingertap-specific [5]. One research system, 4-key Edgewrite [8], does, as it assigns only one key to each finger. However it does not include low-latency as a goal, and still requires aligning fingers and buttons when starting an entry task. Gesture-based text-entry techniques, such as Graffiti, also do not require careful alignment, but even with a stylus, their text-entry bandwidth is a rather slow 20 wpm average after three months of use [1]. Replacing the stylus with a finger is likely slower. Speech recognition can be very fast, but is inappropriate in social contexts where people are expected to be quiet, and error-prone when there is significant noise.

Westerman [9] shows how to track fingers through repeated taps on a touch-sensitive surface. His approach also strove for high-bandwidth over low latency and multitasking, by using a two-handed QWERTY layout. Like Paradiddle, FingerRing [2] supports low-latency one-handed operation using finger-tap detection. However FingerRing is based on rings worn on each finger of a hand, which implies a form

factor that may not be widely acceptable, and combines both sequential and chorded patterns using an explicit timeout that may be difficult to set for both novices and experts.

### IMPLEMENTATION

Our criteria for choosing a prototype implementation for the Paradiddle technique were that it 1) involve physical motions similar to that of our envisioned implementation, 2) not limit the speed at which the user could type, and 3) be easy to build. At first, we intended to use a touch-sensitive Tablet PC, however early tests indicated that the system might limit typing speed because it could not handle multiple accidental simultaneous touches. Though we believe this shortcoming can be overcome with sufficient tuning of hardware and software, we chose to focus instead on evaluating the input technique for our first step. Therefore, to ensure a fast, multi-touch response, we constructed an array of five large (1 sq. in) touch sensors (see Figure 1) connected to a PC through a MakingThings Teleo™ module. This implementation gives the user a limited amount of freedom to move their fingers and presents the illusion of tapping on a flat surface. Because users' hand sizes vary, the hardware prototype allowed different touch sensor patterns to be used for each user (see Figure 2).

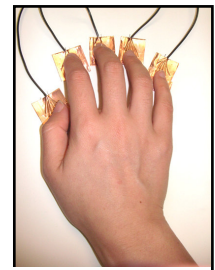


Figure 1: Sensor layout used in our

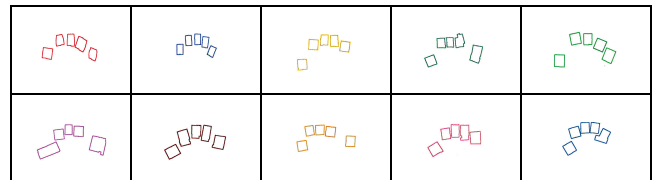


Figure 2: Finger arrangement variation observed in the Paradiddle user study

To help users learn the keyboard tap pattern, Paradiddle uses a visual aid similar to Figure 3. Each character requires three taps, giving a keystrokes-per-character (KSPC) metric of 3. A lower KSPC figure could be achieved by varying the number of taps per character according to letter frequency, but we chose this design to achieve the learning benefits of both alphabetic ordering [7] and taps-per-character consistency [3]. A tap with the fifth finger

A	B	C	A	B	C
111	112	113	141	142	143
D	E	F	D	E	F
121	122	123	121	124	123
G	H	I	G	H	I
131	132	133	131	132	134
J	K	L	J	K	L
211	212	213	214	212	213
M	N	O	M	N	O
221	222	223	241	242	243
P	Q	R	P	Q	R
231	232	233	231	232	234
S	T	U	S	T	U
311	312	313	314	312	313
V	W	X	V	W	X
321	322	323	321	324	323
Y	Z		Y	Z	
331	332	333	341	342	343

Figure 3: Two sequence-to-character mappings used in Paradiddle

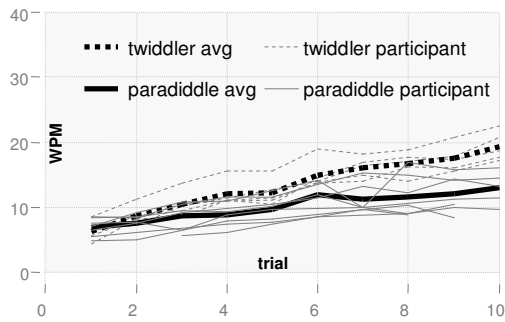


Figure 4: Entry speed across ten trials

performed a backspace.

In addition to a basic mapping (Figure 3, left), Paradiddle also supports a “repeat tap” option (Figure 3, right). In these patterns, a “4” (ring finger) can be substituted for a repeated taps. For example, when the user taps “141,” the “4” on the second digit repeats the “1” on the first digit and registers an “A” as though the user had tapped a “111”. We expected that some users would use this option to achieve higher typing speeds.

#### USER STUDY

We conducted a user study to measure Paradiddle’s typing performance. We chose Twiddler as a comparison point because it also strives for low-latency text-entry. In addition to measuring text-entry bandwidth, we also sought to determine the acceptability, fundamental usability, and learnability of the Paradiddle technique. The study was conducted with 15 English speakers, all of whom happened to be right-handed. All were either staff members or interns at PARC (Palo Alto Research Center). None had used Paradiddle before and two who had briefly used Twiddler were assigned to the Paradiddle condition.

The experiment followed a between-subjects design, in which five participants used Twiddler, and ten used Paradiddle. Of the ten Paradiddle users, five used a version with slightly modified error-correcting capabilities, but these did not turn out to make any significant difference in results. Two participants assigned to Twiddler were expert multi-tap users who frequently sent SMS messages.

Each user performed ten learning trial sessions, each of which lasted about 20 minutes. The trials were spread out over a 1.5 week period. During each trial, a user typed 25 English sentences adapted from the MacKenzie phrase set [5]. The sentences were presented in the context of an animated game, which also showed the map in Figure 3. Users were also interviewed before and after the set of trials, and a focus group discussion was conducted with a subset of users.

#### RESULTS

Figure 4 shows the learning rates for the Twiddler and Paradiddle conditions as a function of trial. The wide lines indicate averages for each condition, and the narrow lines

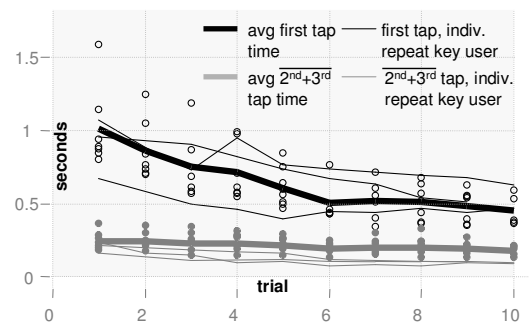


Figure 5: Inter-character tap timings of users who did and did not take advantage of the repeat key option

show performance for individual participants. After 10 trials, Paradiddle users’ average input rate was 13.0 wpm, as compared to 19.3 for Twiddler. The figure for Twiddler is comparable to the speed measured by Lyons, et al. after 10 trials [4]. Paradiddle users made around 0.087 corrections per character compared to 0.027 for Twiddler.

We fit a linear mixed-effects model for typing speed using fixed effects for trial, the Twiddler or Paradiddle condition, and their interaction, and random effects for subjects. Highly significant ( $p < 0.001$ ) effects were found for trial and the interaction between trial and condition, but not for the condition ( $p = 0.775$ ). These results indicate that users learn Twiddler faster (significant difference in slopes), but start at the same point (no significant difference in intercepts).

Three participants chose to use the “repeat tap” feature. Figure 5 shows the benefits of this mechanism. Two sets of data are presented. The upper black lines and points show the average time between the last tap of the previous letter and the following tap of the next. The lower, grey lines indicate the average of both the average time elapsed between the first and second tap and the average time elapsed between the second and third tap. The points mark the averages for a particular trial for all participants who did not use the “repeat tap” feature, the thick line indicates the average over all participants, and the thin lines indicate the performance of users who did use the “repeat tap” feature. The black lines indicate that “repeat tap users” perform about the same as other users in recalling the tap sequence, but by the final trial could perform the second and third taps in about half the time of the non-“repeat tap” users. This result indicates promise for the “repeat tap” feature.

#### DISCUSSION

Although the quantitative results showed that Paradiddle was not as fast or as easy to learn as Twiddler, we believe that the results did show a great deal of promise. For situations demanding text entry for multitasking, particularly when one’s hands need to be available for other activities, a surface-based method like Paradiddle holds advantages that a handheld device like the Twiddler does not. More study is needed in order to understand how to weigh this tradeoff between input speed and manual

freedom. However, the fact that Paradiddle is within range of an established method indicates that Paradiddle can be an attractive option for certain situations. Another advantage of Paradiddle is its adaptability to various finger position preferences, and Figure 2 shows that people took advantage of the flexibility.

### Improvement to the Prototype



**Figure 6: An alternative prototype developed to reduce alignment errors.**

Despite our efforts to avoid using buttons, we ended up using sensors that have one of the worst features of buttons—that they are generally small targets that need to be located and hit accurately—without one of the main advantages—that they provide tactile feedback. Indeed the misalignment due to this factor was the cause of a large number of errors and slowdowns. The

misalignment and discomfort were also caused by ergonomic flaws, including the bulk of the box that housed the electronics and the fact that the fingers needed to be kept elevated when not being used to tap. We believe that this problem will be mitigated in our target situation where users are not sedentary and hands are usually oriented toward the ground. Thus an implementation in a portable or wearable form factor that reduces the need for finger alignment may allow better performance than did our prototype.

As an alternative prototype, we built a compact version on a curved surface, which is expected to reduce the stress on the wrist and reduce misalignment (Figure 6). Misalignment is minimized by easy anchoring for the hand and the stable position of the finger with respect to the sensors. Although the handheld device which requires grasping and releasing does not fit our target form factor, this form factor has allowed us to explore what gains in efficiency could be realized if the problems associated with fingers misaligning with sensors could be alleviated. One of the authors was able to achieve a steady rate of 25 wpm using this prototype after ten focused trials. Note that this result is not comparable with the results reported in the earlier section because this author had received a great deal of exposure to Paradiddle. Nevertheless, this result is suggestive that a relative expert can achieve substantially higher rates, if using a prototype that is designed to reduce errors.

### Improvement to the Mapping

As described earlier, users who employed the repeat key exhibited intra-character tap delays of half the average of the non-repeat-key users. However, this did not translate into a large efficiency gain, because the time-to-first-tap dominated intra-character tap time. We take this result to mean that even after 10 trials, cognitive factors matter more than physical factors. This means that improving learnability, and moving users more quickly towards expert

performance, will be a critical factor in promoting the acceptance of Paradiddle. One approach to this is to select a mapping that facilitates easy memorization. Some previous text entry methods have explored the use of letter shapes to aid the memorability and “guessability” of the mappings [8], and this has been shown to be effective. We plan to explore what types of mnemonic aids might be applied to the arc-shaped layouts that are formed naturally when fingers are laid on a flat surface

### CONCLUSION

For situations involving mobile multitasking, the amount of time it takes to begin and end a brief text entry session may be more important than the speed at which text can be entered. We have presented Paradiddle, a surface-based one-handed text-entry method that is designed to play an important part in a low-latency text entry system because it can be implemented on commonly available hardware and reduces the effort required to align ones fingers with the input device. We tested Paradiddle by comparing it with Twiddler. Although the results show that Twiddler is faster to learn, exploratory modifications show promise for improving Paradiddle’s performance.

### REFERENCES

1. Fleetwood, M.D., Byrne, M.D., Centgraf, P., Dudziak, K., Lin, B., and Mogilev, D. (2002). An analysis of text-entry in Palm OS: Graffiti and the Virtual Keyboard. *Proceedings of the Human Factors and Ergonomics Society 46<sup>th</sup> Annual Meeting*.
2. Fukumoto, M. and Tonomura, Y. “Body coupled FingeRing”: wireless wearable keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing*. CHI '97.
3. Ingmarsson, M., Dinka, D., and Zhai, S. 2004. TNT: a numeric keypad based text input method. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '04.
4. Lyons, K., Gane, B., Starner, T., and Catrambone, R. Improving Novice Performance on the Twiddler One-Handed Chording Keyboard. *Proceedings of the International Forum on Applied Wearable Computing*. March 2005.
5. MacKenzie, I.S, and Soukoreff, R.W. Text Entry for Mobile Computing : Models and Methods, Theory and Practice. *Human-Computer Interaction*, 2002, Vol17, No 2&3, pp 147-198.
6. MacKenzie, I.S, and Soukoreff, R.W. Phrase sets for evaluating text entry techniques. In CHI '03 extended abstracts.
7. Smith, B. A., & Zhai, S. (2001). Optimised Virtual Keyboards with and without Alphabetical Ordering – A Novice User Study. *Proc. INTERACT'2001 - IFIP International Conference on Human-Computer Interaction*.
8. Wobbrock, J., Myers, B., and Rothrock, B. 2006. Few-key text entry revisited : mnemonic gestures on four keys. In *Proceedings of the SIGCHI Conference on Human Factors in Computing*. CHI '06.
9. Westerman, W. Hand tracking, finger identification, and chording manipulation on a multi-touch surface, Doctoral dissertation, University of Delaware, 1999