2012 IEEE International Conference on Systems, Man, and Cybernetics
October 14-17, 2012, COEX, Seoul, Korea

Rapid Screening of Keyboard Layouts

Joonseok Lee

Georgia Institute of Technology
Georgia, USA
joonseok2010@gmail.com

Hanggjun Cho, R. I. (Bob) McKay

Seoul National University
Seoul, Republic of Korea
hanggjun, rimsnucse@gmail.com

***Abstract*—We present a rapid screening method for keyboard layouts. The method relies on user familiarity with a pre-existing layout, and can provide rapid, low-cost estimates of the eventual user speed on the new layout. It provides an effective pre-screening for the expensive training curve estimates of previous methods, and provides valuable supplementary information not available from them. We provide preliminary validation of the method on a mobile phone layout problem.**

***Index Terms*—keyboard, layout, comparison, evaluation**

I. INTRODUCTION

Over time, human factors analysts have devised protocols for evaluating newly-proposed layouts for pre-existing keyboards. These protocols rely heavily on training individuals on the new layout, plotting a curve of performance improvement with training, and using this to predict the eventual performance. These methods are effective, but slow and expensive, typically requiring tens of hours of training, spread over a few weeks, and for a sample of tens of people. In industrial practice, such protocols may be unaffordable unless there is a high certainty that the investment is worthwhile, and generally preclude comparison of a range of possible alternatives.

In many cases, users are already familiar with a pre-existing layout. This situation, of multiple layouts for a single language on a single device, is fairly common: for English, QWERTY and Dvorak layouts are well-known; Chinese has at least five competing layouts for QWERTY-like keyboards, and in our own case, South and North Korea have separate, competing layouts. We propose a pre-screening method, relying on users' pre-existing familiarity with one layout, to quickly and simply estimate their likely eventual speed with new layouts. We demonstrate it with a comparison of alphabetic layouts for English-language input on numeric mobile phone keypads.

II. BACKGROUND

Over the last two decades, many authors have suggested new layouts for various kinds of character input device. They usually evaluated their validity through user study. The most widely-used methodology was to ask a group of people to type standardized text using the new layouts and a standard one, and compare the elapsed time. Many authors tried to deal with the difference in familiarity through training, but this raises the cost of testing, since training to attain basic level proficiency takes many hours over a substantial period.

For comparing rapid testing protocols, we set several criteria which a protocol should satisfy:

1) Effects of differences in familiarity should be minimized.
2) Comparisons should be at near-expert level if possible.
3) Testing should minimize the time cost to participants.
4) Testing should be extensible to many varied testees.

Previous strategies can be roughly categorized into four groups: ignoring the issue, restricting testing to novices, providing training session, and observing learning progress.

1) Ignoring the Issue: one constructs participant groups independent of familiarity. This may be unavoidable when the physical design is new. A number of researchers have used this approach to analyze initial acceptance of new designs, and were able to conclude that it was acceptable even if it was significantly slower than the control. The method has been used for evaluating a pen-based computer [1] and a mobile phone keypad [2]. It satisfies criteria 3 and 4, but ignores 1 and 2.

2) Training Sessions: this strategy allows a little time for familiarization on new layouts before testing. Typical designs restrict the familiarization to the level that criteria 3 and 4 are still acceptably satisfied, while performance on criterion 1 is improved; however criterion 2 is not satisfiable. This approach has been used for evaluating a mobile phone keypad [3], [4].

3) Restricting Testing to Novices: conversely, we can remove the effects of familiarity by testing participants who are completely unfamiliar with all layouts. It eminently satisfies criterion 1. However, it may be difficult to find such naive users (e.g. people who have never used a numeric keypad), and such a sample may be biased, so there are problems with criteria 3 and 4. Moreover this method gives little indication of expert-level performance, failing criterion 2. Nevertheless, its simplicity has led to previous applications such as [5].

4) Detailed Observation of the Learning Progress: the 'gold standard' provides sufficient training time for substantial progress. Once the learning rate decreases, speed is plotted against training time, from which the participant's attainable typing speed can be estimated. This method has been widely used [6], [7]. However, it fails criteria 3 and 4, because of the very extensive training time (perhaps a week of participants' time), and thus is poorly suited as an initial screening test.

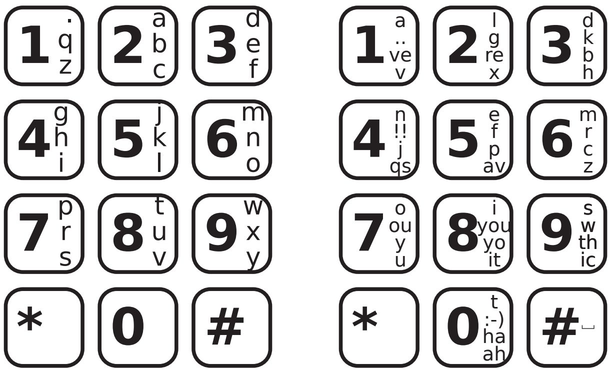
III. METHODOLOGY

A. Familiarity Issues

There are two familiarity issues in comparing keyboard layouts. Firstly, participants may differ in their conscious recollection of key locations in a familiar layout such as QWERTY,

978-1-4673-1714-6/12/$31.00 ©2012 IEEE

whereas they will all be unfamiliar with a newly proposed layout. A second familiarity category is finger memory. We build physical memory of the keyboard sequences of frequently used phrases. Experienced users do not need to separately locate the ‘t’, ‘h’, and ‘e’ to type ‘the’. We need to eliminate biases arising from both kinds of familiarity.



(a) ABC Layout (b) PM Layout

Fig. 1. Standard and Personalized Multigram-based Keypad Layouts [8].

To deal with such familiarity issues and remove the bias, there are two alternatives: standardizing to the higher level of an experienced user; or to the lower level of a novice. In our strategy, we use both: remapping all layouts via symmetries to an unfamiliar layout (to cancel out finger memory), but then specifying the string to be typed in terms of the well-known layout (to take advantage of user knowledge of that layout). For clarity, we explain these transformations in the reverse order. As a running example, we use an evaluation on keypad layouts; specifically, we will evaluate the expected efficiency of a keypad layout optimized for a specific user’s English-language cellphone SMS usage (the PM layout [8]). Our test users were familiar with an alphabetic layout widely used on Samsung phones in Korea (the ABC layout). Both layouts are shown in Fig. 1; a detailed rationale for the PM layout is presented in [8]. However we emphasize that this comparison is for illustration purposes only: the comparison methodology is general, and could just as readily be applied, for example, to comparison of QWERTY and Dvorak layouts for English, or South and North Korean layouts for Korean (Hangul). For brevity, the methodology is explained informally in this paper; a detailed mathematical formalization is available in [9].

B. Character Mapping: Handling Location Memory

We want a quick estimate of users’ expected proficiency, after long-term use, of new layouts. But we have no such users. How can we compensate for the users’ unfamiliarity with the new layouts? Suppose that we want the user to type the word ‘compile’. In the ABC layout, the user should type the sequence ‘222.666.6.7.444.555.33’ (we denote multiple key presses by repeating the key, and use a period to denote the pause between characters). In the PM layout, they should instead type ‘666.7.6.555.8.2.5’. But they already know the former, and will type it rapidly, whereas they will have to search to type the latter. The solution is straightforward in this case: they can be requested instead to type ‘opmltaj’ in the ABC layout. This nonsense word has no meaning in ABC, but typing it will produce the same string, ‘666.7.6.555.8.2.5’, as would be required to type ‘compile’ in the PM layout, with exactly the same finger movements.

The alert reader will have noticed some complications:

1) PM layout uses the ‘0’ key, which is unused in ABC
2) PM layout uses four strokes, rather than ABC’s limit of three, for some characters (for example, ‘computer’ is represented by ‘666.7.6.555.7777.0.5.66’ in PM)
3) PM uses multigrams, so ‘your’ could be typed as ‘777.7.7777.66’ or as ‘777.77.66’ or as ‘88.7777.66’.

These issues do not arise in most comparisons (e.g. QWERTY vs Dvorak), so for the moment we only consider phrases where these complications do not occur. Of course, there are remaining issues relating to the relative familiarity of ‘compile’ and ‘opmltaj’ and finger memory of ABC sequences such as ‘ers’ by comparison with the corresponding PM-to-ABC sequence ‘jnw’; we deal with these issues next.

C. Layout Transformation: Removing Finger Memory Bias

Our methodology handles the greater familiarity of ‘compile’ than ‘opmltaj’ by transforming them to a pair of equivalent, but equally unfamiliar, words. It relies on the symmetries of the keypad. Most physical keypads have symmetries that only very weakly affect typing speed. For example, typing speed on QWERTY layouts may be little affected by reflections about the vertical axis, and to a somewhat greater extent by reflections about the central alphabetic row. Of course, in each such case, we cannot simply assume invariance, but need to experimentally validate it. In the case of the ABC keypad, we rely on reflections through the central axis, as shown in Fig. 2. We conjecture that these reflections will not severely affect typing speed (except for typists with asymmetric disabilities), since they merely exchange which hand makes specific movements, and reverse the direction of movement.

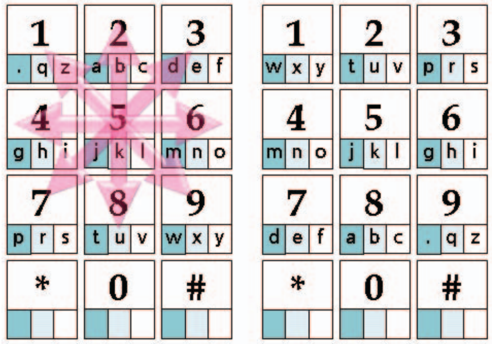


Fig. 2. Transformation of ABC Layout
(Left: Reflections, Right: Transformed Layout)

Now, to compare the layouts for the string ‘compile’, we reflect both ‘compile’ and ‘opmltaj’ through the central axis, generating ‘vigdolr’ and ‘idglatj’ – equally nonsensical, yet still (if our proposed typing invariants hold) of equal typing difficulty as the corresponding ABC and PM strings.

Before starting the detailed experiment, it is important to check whether the layout transformation leads to biases. Two kinds of check are needed:

1) Are the proposed invariants (finger frequencies, row frequencies etc.) preserved in the target language? This can be checked computationally.
2) Does the transformation significantly affect typing speed? This requires experimental verification.

The former check is simple: sample the test corpus in original and transformed forms, collect statistics of the invariants, and confirm whether they are preserved.

But we might have been wrong in our conjectures regarding the invariants. In our case handedness, or cramping from typing on the bottom row, may affect speed more than we have assumed. Experimental validation is required: measuring whether transformations preserving the invariants also preserve typing speed (i.e., we ask testees to type the same phrases before and after transformation). In general, we won't get perfect invariance – there will be some effect from swapping hands or exchanging rows – which gives us three main options:

1) If the bias is large, discard the invariants and start over.
2) If the bias is small relative to performance differences, ignore it.
3) If the scale of the bias is intermediate, correct the (now known) bias in comparing typing speeds.

D. Familiarity Difference

Although this procedure has removed systematic biases, bias can still arise from small-sample effects. Suppose one layout generates 'uatspn', another 'wxkjlq'. Both are nonsense, but the first is much more 'English' in feel than the latter. With feasible sample sizes, such examples may add significant noise to our results. To reduce this, we post-prune our transformed corpuses, effectively stratifying them by removing samples if their letter-level frequencies differ too much.

Formally, we define the letter-frequency familiarity ($\mathfrak{F}$) of a string $s = (s_1, \ldots, s_n)$ of length n over a set L of letters:

Definition 1.

$$\mathfrak{F}(s) = \frac{1}{n}\sum_{k=1}^{n}\frac{p_T(s_k)}{p_U(s_k)} \quad (1)$$

$$= \frac{|L|}{n}\sum_{k=1}^{n}p_T(s_k) \quad (2)$$

p_T is the frequency of each letter in the target alphabet
p_U is the corresponding uniform distribution
s_k is the kth character of string s

Thus $\mathfrak{F}$ is a measure of the excess frequency of the string over what we would expect from a uniform distribution.

Next, we define the bias $\mathfrak{B}$ of a pair of strings (s_1, s_2) as:

Definition 2.

$$\mathfrak{B}(s_1, s_2) = |\mathfrak{F}(s_1) - \mathfrak{F}(s_2)| \quad (3)$$

The value of $\mathfrak{B}$ lies in the range $[0 \ldots |L|]$, with smaller values corresponding to fairer comparisons.

E. Overall Experimental Protocol

Putting all these together, we propose a protocol for comparing new key layouts with a familiar layout. We illustrate with comparing the Dvorak and QWERTY layouts. We first generate a list of words from an appropriate corpus. We must also define the keyboard transformation. Here we use left-right hand symmetry, and vertical symmetry of the letter area. Thus, we interchange horizontal columns leftmost-to-rightmost, then exchange rows 1 and 3. We convert each element into the new and old keyboard layouts, apply the reflections to them, and finally represent each as the corresponding QWERTY keystrokes.

This gives us a tuple of strings, one for each layout. We check the bias of each tuple, eliminating those above our threshold, to generate the test set we expose to testees.

IV. EXPERIMENTS: COMPARING PM AND ABC LAYOUTS

We now detail the experimental comparison between PM and ABC layouts, both to show how the protocol can be extended to deal with various complications, and to demonstrate the kinds of information that can be obtained at low cost.

A. Applying the Test Protocol

1) Defining the Invariants: For the 12-key mobile phone keypad, we used as invariants:

1) the number of strokes required for each letter
2) the frequency of consecutive uses of the same key
3) the frequency of consecutive uses of the same hand
4) the average distance of movement of the thumbs
5) the balance between hands
6) the distance from each key to the center

2) Layout Transformation: If it were not for the complications raised in subsection III-B, the simple reflection-through-the-center transformation we previously defined would be sufficient: it preserves all the invariants listed above. So how should we handle these complications?

1) Use of 0 key: how should we reflect 'computer', which is '666.7.6.555.7777.0.5.66' in PM? We decided to reflect '0' to '2', so that after reflection it becomes '444.3.4.555.3333.2.5.44', i.e. the ABC string 'idgl?ajh'.
2) Extra strokes: what should we do about the 'u', which became '3333' on reflection? Our solution is to split it into a pair of characters, , i.e. '444.3.4.555.33.33.2.5.44', represented in ABC by the string 'idgleeajh'.
3) Multigrams: we assumed that a proficient PM user would reliably choose the most efficient multigram for each word, so that is the representation we test (for example, we would choose '777.77.66' for 'your', since it is the most succinct representation.

B. Experimental Design

We enrolled a total of 116 participants (60 male and 56 female). Their ages ranged from 18 to 60. We did not directly control for previous experience with the ABC layout, since this would have required a further round of detailed measurement. Subjectively, some were highly skilled, while others had more

Please type your name:
Joonseok
○ M ● A ○ B ○ C ○ D
Please type the word:
dwlpth
dwl
Progress: 0/50
Elapsed time: 0

Fig. 3. Phone Model (Left) and User Interface (Right) used in the Experiment

limited experience, but all had some. We directly measured typing speed as a proxy for their ABC layout experience.

This test was conducted only once for each participant. We did not need or allow any training session, because the methodology directly uses whatever is their current facility in the ‘old’ (ABC) layout, and internally equilibrates any differences in familiarity. We instructed each participant to type as fast and accurately as possible, and not to pause while typing a word. We gave no hint about how the test strings were constructed, describing them as ‘50 random strings’.

For the text corpus, we used an archive of mobile phone messages from the first author.[1] We dropped special characters, which would require complex lookup in the ABC layout (and for most, in PM as well). Since PM does include some special characters, this results in some bias against PM. We also excluded words of length less than 3 or more than 10. Of the 554 words originally extracted from the archive, 178 remained after filtering for bias. For each participant, a randomly chosen 25 words (i.e. 50 strings in total for the two layouts) were to be typed. To remove any possible sampling bias, we randomized the order of words. For each pair and each participant, we also randomized whether the word derived from PM or ABC layout was presented first.

We used a programmable phone (Samsung SCH-M470) with an ABC-layout physical keypad. Fig. 3 shows the phone and user interface of our test software.

C. Results and Analysis

1) Overall Results: We measured the elapsed time for typing each word during the test. To estimate the overall improvement from the ABC to PM layout, we averaged the elapsed time for strings generated by the ABC and PM layouts. We then applied a correction to compensate for the 4-stroke to 3-stroke conversion described earlier. As we see from Table I, participants took 19.32% less (corrected) time to type PM strings than ABC. The biggest improvement was 34.86%; only 2 out of 116 participants recorded a lower performance with the PM layout (-1.43% and -10.52%). We tested the difference between the two layouts using a Wilcoxon Rank Sum Test [10]. We found a p-value of 0.00000025 for the hypothesis that the PM layout is not faster than the ABC layout – that is, we can accept that the PM layout is faster with more than 99.9% confidence.

TABLE I
AVERAGE AND STANDARD DEVIATION OF ELAPSED TIME FOR ABC AND PM STRINGS (SECONDS)

	Elapsed Time
ABC Strings	8.16 ± 2.58
PM Strings	6.53 ± 2.02
PM Strings (Corrected)	6.58 ± 2.02

TABLE II
IMPROVED, SIMILAR, AND WORSENED WORD PAIRS WITH PM LAYOUT, BY THE AVERAGE ELAPSED TIME OF EACH PARTICIPANT.

	Improved	Similar	Worse
Count	16.34 ± 2.45	2.36 ± 1.56	6.30 ± 2.18
Percentage	65.34%	9.45%	25.21%

For another perspective, we counted how many strings gave substantially improved (PM is faster than ABC by more than 10%), similar or worse (PM is worse than ABC by more than 10%) performance with PM layout. From Table II, we see that about two thirds of pairs show better performance in the PM layout, but a quarter are worse.

2) Effect of Familiarity: To test our argument that this protocol would discount the effects of familiarity with the ABC layout, we compared the level of improvement between ABC and PM layouts, for each 10-person-range of participants, ranked by speed. The speed is calculated from the total time, including both ABC and PM strings. Table III shows fairly stable improvement from ABC to PM layout in all groups: PM is faster than ABC by around 20%, independent of ABC speed, even though there is a very large difference in the raw speed itself. Since we see little difference, it seems that the protocol has successfully eliminated the bias resulting from the familiarity difference.

TABLE III
MEAN ELAPSED TIMES BY ABC FAMILIARITY LEVEL (SECONDS)

Speed Rank	Count	ABC	PM	PM (Corrected)	Improvement
1 - 10	10	4.34	3.51	3.57	17.85%
11 - 20	10	5.47	4.42	4.47	18.26%
21 - 30	10	6.11	4.80	4.85	20.61%
31 - 40	10	6.51	5.44	5.50	15.49%
41 - 50	10	7.32	6.10	6.16	14.81%
51 - 60	10	8.08	6.26	6.32	21.72%
61 - 70	10	8.22	6.55	6.61	19.62%
71 - 80	10	8.77	6.97	7.03	19.82%
81 - 90	10	9.40	7.54	7.60	19.13%
91 - 100	10	10.39	8.50	8.54	17.75%
101 - 110	10	11.74	9.11	9.16	21.97%
111 - 116	6	13.97	10.85	10.91	21.95%

---

[1] This archive was actually used to define the PM layout; for a detailed discussion of why we view personalized layouts as useful, and how the PM layout was derived from the archive, please refer to [8]

TABLE IV
PREDICTED TYPING SPEED IMPROVEMENT FOR PM LAYOUT UNDER DIFFERENT METHODOLOGIES

No.	Text Filtering	Test Method	Improvement
1	No	Theoretical Analysis	45%
2	No	User Study (direct training)	53%
3	Yes	Theoretical Analysis	28%
4	Yes	User Study (our protocol)	19%

V. DISCUSSION

The principal aim of our methodology is to measure differences in expected physically-limited typing performance between layouts, and to do so with low per-participant experimental effort. This would reduce the cost of testing of new layouts, and/or enable a much wider sample of participants, eliminating a major source of bias. So how well does it stack up against the classical methodologies?

First, we need to consider the differing cognitive loads in the two protocols. In the classical training approach, the text typed by the participants is real English words – hence easy to remember. In the new protocol, the strings look random. During the "Character Recognition" phase – the first phase in the process of typing [11] – the eye does not read with maximum achievable speed (the required speed for comprehension). Instead, the eye reads the characters just fast enough to feed the copy to the hand as needed [12]. In the next phase, remembering the characters to be typed in short-term memory, we can buffer just 4 to 8 letters, preventing further look-ahead [13]. Overall, random-appearing strings may be much more difficult to remember because of unfamiliarity. Participants in our experiment confirmed that they had to check the next character repeatedly during typing. This inevitably reduced their overall performance, but to a greater extent for faster typing, thus reducing the overall difference in performance. This is a systematic issue with the new protocol – it is inherently conservative, and will underestimate any performance differences between new and old layouts. While it would be desirable to eliminate such biases, at least it is a bias in the right direction.

Second, in a traditional evaluation, the familiarity difference (between new and old layouts) is handled by learning curves. Participants are assumed to be familiar with the standard layout. After a few sessions of training, perhaps we see performance with the new layout coming close to that with the old. In such cases, we can conclude that the new layout is faster. However rapid improvement during initial training (several hours to several weeks) may not imply higher ultimate speed. It is possible that the new layout is just easier to learn, rather than more efficient. Conversely, a new layout which performs poorly in the initial stages may be better than the standard after more training. In other words, the extrapolation required by the classical approach is substantial, and may be wrongly estimated.

For example, Strong compared typing speeds for QWERTY and Dvorak [14]. For fair comparison, he trained 10 typists with Dvorak until they attained their previous QWERTY speed. Then he invited 10 more typists who were familiar with QWERTY layout. Each group was trained with each layout for a further 100 hours, and Strong recorded the learning curves. Surprisingly, the Dvorak group did not show better performance than the QWERTY group with further training, even though Dvorak is reportedly quick to learn [5], and Strong's participants did learn it quickly. Thus we cannot simply assume that initial stage data can be a good estimator for ultimate speed. [2]

TABLE V
MODEL-BASED FITNESS VALUES, FAMILIARITY ($\mathfrak{F}$) AND ELAPSED TIME (SECONDS)

	Fitness Value	Familiarity ($\mathfrak{F}$)	Elapsed Time
ABC Strings	2.26 ± 0.49	1.24 ± 0.38	8.14 ± 3.60
PM Strings	1.56 ± 0.42	1.25 ± 0.50	6.51 ± 3.28

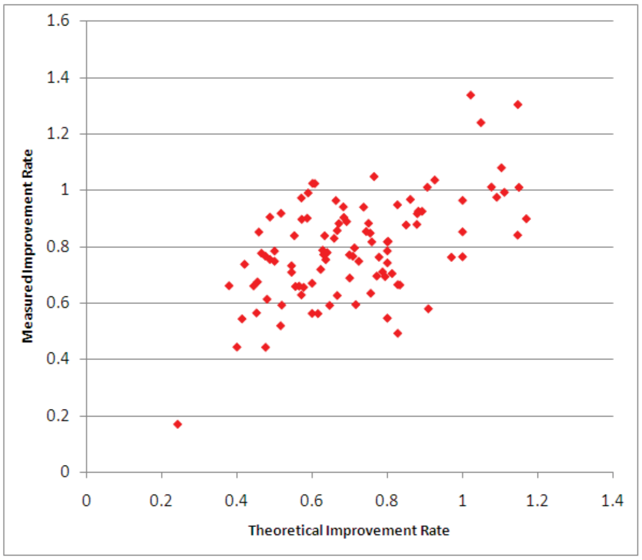


Fig. 4. Measured Improvement Rate vs. Theoretical Estimates for each word

Third, there is the issue of statistical stability. In [8], we reported a classical comparison of ABC and PM layouts. As is common, the results were based on a sample of only ten, too small for statistical reliability.Moreover seven of ten participants were male Computer Science graduates, whose performance may be atypical. The new protocol is able to avoid both sources of bias because of the low cost of testing.

3) Results for Individual Word Pairs: In Table II, approximately a quarter of words were slower to type with PM layout than with ABC. Can we explain this? We investigated various possible explanations for the differences:

1) They may result from the physical motions, hence be predictable from a physical model.
2) They may be due to differences in familiarity ($\mathfrak{F}$), hence be predictable from differences in $\mathfrak{F}$.
3) They may be due to other causes.

---

[2] This experiment may be flawed in another way. Every participant had experience with QWERTY, so the Dvorak group may have been handicapped by the confounding effect of QWERTY [15].

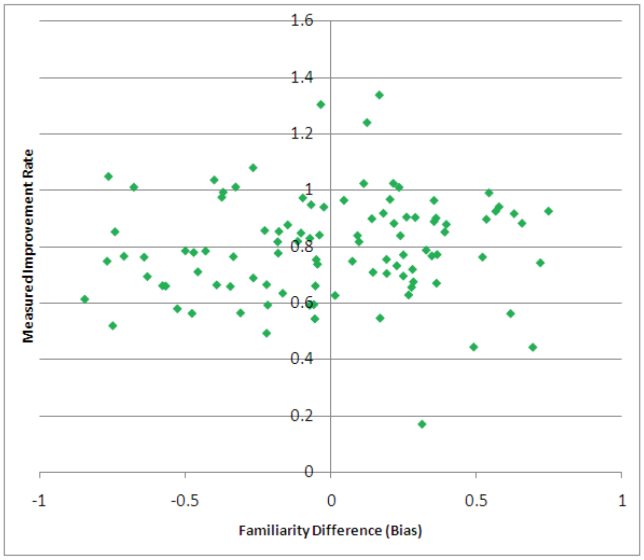


Fig. 5. Measured Improvement Rate vs. Familiarity Difference for each Word

We can verify the first two by computing the relevant differences (fitness, as defined in the physical model from [8], and familiarity $\mathfrak{F}$), and comparing to the differences in elapsed time. The overall statistics are shown in Table V. From Figures 4 and 5, we can see that there is a reasonable correlation (0.53) between the measured and model-based estimates of improvement, so that a substantial portion of the difference could result from the physical differences in typing. Notably, the quarter of strings slower in PM are fully explained by the physical model. On the other hand, there is no correlation (0.01) between differences in $\mathfrak{F}$ and performance. We cannot exclude other influences (since the correlation with fitness was not 1.0), but it is possible that much of the remaining variation is due to sampling and measurement errors.

VI. CONCLUSION

We have proposed a rapid methodology for comparing new key layouts with a previous familiar one. It offers three main advantages over earlier direct-test comparison methods: 1) a fairer comparison of layouts at the expertise level of the familiar layout, 2) widening the range of participants, and 3) reducing experimental cost in time and equipment.

However, we are relying on some assumptions. One is that we are comparing layouts for exactly the same keyboard. However we saw that we can compensate for slight differences (use of ‘0’ key in PM). We also assume that we can identify a set of suitable invariants, and symmetries preserving those invariants. Since the invariants will rarely be exact, we need to quantify the failures, so that we can either compensate for them, or at least estimate their maximum effect on the results. We also assumed that letter distributions on the keyboard are unaffected by these symmetries. This may not always be the case (as, for example, with the S. Korean keyboard, which has consonants on the left, vowels on the right).

The proposed method dramatically improves the cost of comparison, but it suffers from some limitations. It gives no information about learning effort. Also, it tends to underestimate long-term performance due to cognitive load. Thus it is complementary to training-based methods. It can be effectively used for fast screening of new keyboard layouts in a multi-protocol evaluation: only keyboards which pass this validation would be evaluated with a training-based protocol. The initial screening effort is not wasted, since it provides supplementary information about the mechanical limits to speed that it is not feasible to get from training-based protocols. Finally, it can provide validation across a wider range of testees than is feasible for training-based methods.

ACKNOWLEDGMENT

This work was supported by a National Research Foundation of Korea (NRF) grant funded by the Korean government (MEST) Project No 2012-004841. The Institute for Computer Technology of Seoul National University provided research facilities for the study. We deeply appreciate the effort of all volunteers in our human evaluations.

REFERENCES

[1] I. S. MacKenzie, R. B. Nonnecke, J. C. McQueen, S. Riddersma, and M. Meltz, “A comparison of three methods of character entry on pen-based computers,” *Human Factors and Ergonomics Society Annual Meeting Proceedings*, vol. 38, no. 5, pp. 330–334, 1994.

[2] A. Mittal and A. Sengupta, “Improvised layout of keypad entry system for mobile phones,” *Computer Standards and Interfaces*, vol. 31, no. 4, pp. 693–698, 2009.

[3] L. Butts and A. Cockburn, “An evaluation of mobile phone text input methods,” *Aust. Comput. Sci. Commun.*, vol. 24, no. 4, pp. 55–59, 2002.

[4] J. Gong and P. Tarasewich, “Alphabetically constrained keypad designs for text entry on mobile devices,” in *CHI ’05: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA: ACM, 2005, pp. 211–220.

[5] J. Harnett, “Qwerty lives another day,” *Office Equipment News*, pp. 15–57, 1972.

[6] S. E. Michaels, “Qwerty versus alphabetic keyboards as a function of typing skill,” *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 13, no. 8, pp. 419–426, 1971.

[7] I. S. MacKenzie and S. X. Zhang, “The design and evaluation of a high-performance soft keyboard,” in *CHI ’99: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA: ACM, 1999, pp. 25–31.

[8] J. Lee and B. McKay, “Optimizing a personalized cellphone keypad,” in *Proceedings of International Conference on Convergence and Hybrid Information Technology*, ser. Lecture Notes in Computer Science, D. H. Geuk Lee and D. Slezak, Eds., vol. 6935. Daejeon: Springer, September 22–24 2011, pp. 237–244.

[9] J. Lee, H. Cho, and B. McKay, “Comparing typing speeds for keyboard layouts fairly and efficiently,” Seoul National University Structural Complexity Laboratory, Seoul, Korea, Tech. Rep. TRSNUSC:2012:001, April 2012. [Online]. Available: https://sc.snu.ac.kr/sclab/doku.php?id=research:techreports

[10] F. Wilcoxen, “Individual comparisons by ranking methods,” *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[11] W. E. Cooper, *Cognitive Aspects of Skilled Typewriting*. New York, USA: Springer-Verlag, 1983.

[12] R. Butsch, “Eye movements and the eye-hand span in typewriting,” *Journal of Educational Psychology*, vol. 23, no. 2, pp. 104–121, 1932.

[13] P. Buzing, “Comparing different keyboard layouts: aspects of qwerty, dvorak and alphabetical keyboards,” Delft University of Technology, Delft, Netherlands, Tech. Rep. 60, 2003.

[14] E. P. Strong, *A Comparative Experiment in Simplified Keyboard Retraining and Standard Keyboard Supplementary Training*. Washington D.C, USA: US. General Services Administration, 1956.

[15] J. Noyes, “The qwerty keyboard: a review,” *International Journal of Man-Machine Studies*, vol. 18, no. 3, pp. 265–281, 1983.