To appear on the International Conference on Human Factors in Computing Systems (CHI 2005)

Predicting Task Execution Time on Handheld Devices Using the Keystroke-Level Model

Lu Luo
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213
luluo@cs.cmu.edu

Bonnie E. John
Human Computer Interaction Institute
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213
bej@cs.cmu.edu

ABSTRACT
The Keystroke-Level Model (KLM) has been shown to predict skilled use of desktop systems, but has not been validated on a handheld device that uses a stylus instead of a keyboard. This paper investigates the accuracy of KLM predictions for user interface tasks running on a Palm OS based handheld device. The models were produced using a recently developed tool for KLM construction, CogTool, and were compared to data obtained from a user study of 10 participants. Our results have shown that the KLM can accurately predict task execution time on handheld user interfaces with less than 8% prediction error.

Author Keywords
Keystroke-Level Model, cognitive modeling, task execution time, handheld device.

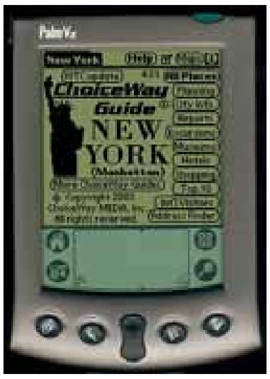
ACM Classification
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

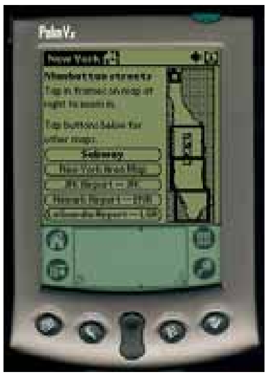
INTRODUCTION
The growing popularity and unique attributes of handheld devices, such as Palm OS® based personal digital assistants (PDA) and pocket PCs, have brought new challenges to user interface design. To achieve higher portability, handheld devices are often equipped with a small touch screen display, a stylus pen, and several hardware buttons for the user to perform tasks. This equipment enables novel interaction methods such as handwriting recognition and gestures, and has different attributes than the display, keyboard, and mouse combination in the conventional command-line or graphical user interfaces. In these cases, it is necessary to estimate task execution time early in the design of handheld systems and a predictive modeling technique would be a useful tool.

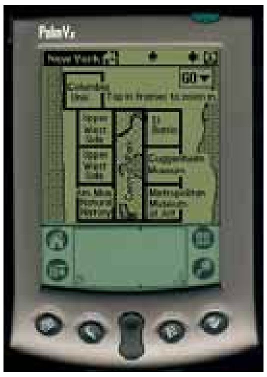
The Keystroke-Level Model (KLM), created by Card, Moran, and Newell in 1980[2], has been applied in many previous HCI studies. The KLM is a simple but accurate means to produce quantitative, *a priori* predictions of task execution time at an early stage in the development process. The scope of the KLM is limited to *skilled users* performing *error-free task* using a *specific method* on a given interface design. The basic idea of KLM is to list the sequence of keystroke-level actions that the user must perform to accomplish a task, and sum the time required by each action. The KLM describes the task execution in terms of four physical-motor operators: **K** (key-stroking), **P** (pointing), **H** (homing), and **D** (drawing), one user mental operator **M**, and a system response operator **R**(*t*). K, P, H and D are determined by the actions necessary to accomplish the task. The KLM assumes that the first five operators take constant time for each occurrence, and provides a set of heuristic rules for placing M's in the sequence of Ks, Ps, Hs and Ds, set by prior psychology and HCI research. Response times must be estimated by the analyst and only include the time that the user must wait for the system after any M operator has completed.

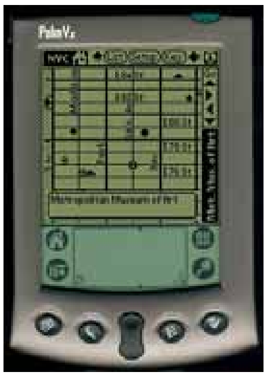
The KLM has been applied to many different tasks such as text editing, spreadsheets, graphics application, and highly interactive tasks [2, 4], but has not previous been shown to work for a stylus-based interface. This paper investigates the applicability and prediction accuracy of the KLM on handheld interfaces. The following sections will first describe the four tasks chosen to be performed on a Palm OS® based PDA. Then, the KLMs constructed using CogTool [5] for each task are given. In order to verify the task execution time predicted by the models, a user study was performed on 10 participants. By comparing the actual time user spent on each task with the corresponding task execution time predicted by the models, we show that the KLM works well for handheld interfaces, with an average prediction error of 3.7%.

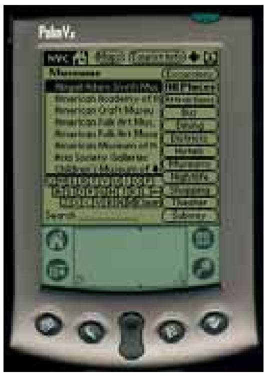
TASK DEFINITION
We started our study by choosing the tasks to be modeled and compared with user data. Based on the availability of equipment and research tools, we used the popular Palm Vx PDA (Palm OS) as the target handheld platform. Pictures of Palm Vx can be found in Figure 1.


(a) Start page of CWG-NYC


(b) Manhattan region map


(c) Detailed region map


(d) Street map


(e) List of museums

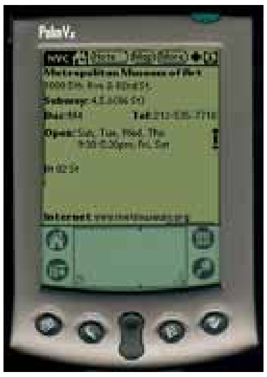

(f) Query result

Figure 1 Snapshots of the CWG for NYC application

The Palm Vx PDA has a stylus pen, several hardware buttons, and a touch screen divided into two areas. The larger area on top is used to display information and allows the user to perform operations by tapping on icons, menus, lists, buttons, and other interface elements drawn on the screen. The smaller area at the bottom has four shortcut icons for quickly opening frequently used functions. In the center of this area in which to write a shorthand, known as *Graffiti*, users can input text to the device by drawing gestures in the Graffiti area.

When choosing the tasks, we kept in mind two principles. First, the operations required to accomplish the tasks should cover as many different interaction methods used in the target platform as possible. Second, for comparison purposes, the tasks should be to accomplish the same goal but using different methods. We looked at over 100 off-the-shelf applications for Palm OS and selected an application named ChoiceWay Guides (CWG). The copy of CWG we purchased for our study is the Palm OS version guide for New York City (NYC). The WinCE version and more details can be found at http://www.choiceway.com/. CWG for NYC allows the user to understand city facts, plan trips, and search for information like open hours and telephone numbers about a particular place. Figure 1(a) shows the start page of the CWG for NYC application. The user can tap on one of the icons displayed on the screen to perform corresponding operations indicated by the icon text.

All tasks in this study share the same goal of *finding the opening hours of the Metropolitan Museum of Art* (MET). Based on the functionality of the application, there are four different methods to accomplish this goal:

Method 1: Map Navigation. From the start page shown in Figure 1(a), tapping on the “Maps” icon at the top right corner will lead to Figure 1(b), which displays Manhattan map divided into three regions. This method requires the user to have some basic knowledge of where the MET is in Manhattan. Tapping corresponding areas in the region map will lead to the detailed region map in Figure 1(c) and the street map in Figure 1(d). Tapping on the spot where the MET is located in the street map displays the name of the MET in a box at the bottom, which leads to the query result shown in Figure 1 (f).

Method 2: Soft Keyboard. From the start page, tapping on the “Museums” icon located at the center right side gives Figure 1(e), an alphabetic list of museums. Using the soft keyboard located at the bottom of the screen, the user can then input the letters “METRO…” one by one. When MET is shown in the list, tapping on the item leads to the query result in Figure 1(f).

Method 3: Graffiti. The only difference from Method 2 is that at Figure 1(e) Graffiti is used to input the letters instead of the soft keyboard.

Method 4: Scroll Bar. At Figure 1(e) the user taps the scroll down arrow at the right of the list of museums until the MET is shown. The user then taps on MET to get the desired information.

MODEL CREATION

The KLM for the four tasks described above were created using CogTool[5], a suite of software tools built to facilitate modelers to quickly produce correct KLMs. CogTool allows the modeler to mock up an interface as an HTML storyboard and demonstrate a task on the storyboard using Netscape web browser. The demonstration events are captured by Behavior Recorder [6], which automatically generates a KLM that includes all Ks, Ps, Hs, and Ms required to accomplish the task. The KLM is implemented in ACT-Simple [7] which compiles into ACT-R [1] code. The task execution time is then calculated by running the generated KLMs in the ACT-R environment.

The HTML mock-ups for the four tasks were generated from the Palm OS Emulator, which emulates the hardware of various models of Palm Powered™ handhelds. The emulator enables a "virtual" handheld device to run on a desktop machine[1]. Given this feature, we installed the CWG for NYC application on the emulated Palm Vx, and took snapshots for all the steps in each task. The pictures in Figure 1 are examples of the snapshots. The snapshots were then used to create the HTML mock-ups. More details about using CogTool can be found in [5, 6].

Figure 2 shows a fraction of the KLM, expressed in ACT-Simple code, generated for the Graffiti task. In this model, the physical-motor operators such as `(press-button Museums)` were captured by the CogTool Behavior Recorder when the task was demonstrated. The `(look-at)` and `(think)` operators were automatically added by CogTool. Task execution times were calculated by running the KLMs in the ACT-R environment.

```
(klm-p (klm-goal klm
(think)
(look-at "Museums")
(press-button "Museums")
(think)
(look-at "-graffiti-")
(press-button "-graffiti-")
(think)
(press-button)
(think)
(look-at "MET")
(press-button "MET")
 … …
```

Figure 2 Example KLM code for Method 3

USER STUDY

To verify the task execution time predicted by the KLMs, we performed a study with 10 expert PDA users. All the participants were college or graduate students who own one of the several kinds of handheld devices, including Palm OS based PDA, pocket PC, and smart cell phone. Although not all of these PDA users were skilled at Graffiti, they all were skilled at gesture-based text entry and the training session (below) allowed them to learn the three Graffiti gestures required for Method 3. Table 1 shows the information of each participant including the gender, the model of PDA owned, and for how long it has been used.

Event Logger

User task execution times were obtained using EventLogger, a Palm OS system extension that records system events to a log file. The log files are Palm database (PDB) files in text format. Each line of a PDB log file is a tab-delimited listing of one system event, in the form of "*TickCount sysEventName OptionalInfo*". The *TickCount* is the time stamp of the event, the *sysEventName* is the name of the event, and the *OptionalInfo* includes information such as the character entered in a keystroke event, the name of the form in a form open event, etc. User execution time for each task can be obtained by calculating the difference between the starting and ending event timestamp, and dividing the difference by the number of system ticks per second defined in the header part of the PDB file.

#	Gender	Device owned (OS)	Time
1	Male	Palm Vx (Palm OS)	5+ years
2	Male	Compaq iPAQ (Win CE)	3 years
3	Male	Palm IIIe (Palm OS)	4 years
4	Male	Handspring Visor (Palm OS)	3 years
5	Female	Handspring visor Pro (Palm OS)	2 years
6	Female	Dell PDA (Win CE)	1 year
7	Male	iPAQ 3630 (Win CE)	4 years
8	Male	Kyocera 7135 (Palm OS)	4+ years
9	Male	Handspring Visor Prism (Palm OS)	3 years
10	Female	Palm VA (Palm OS)	3 years

Table 1 Information of participants

User Study Process

Each participant was first asked to practice the tasks in a training session, followed by the actual session where the participants were asked to perform the four tasks 10 times each. In the training session, participants were asked to carefully read the step-by-step instructions on how to operate the EventLogger and how to perform the tasks. The participants were required to strictly follow all steps and repeat each task for 10 times. During this training session, the participants were told to focus on becoming familiar with the tasks. The PDB files from this session were saved as training data. In the second session, the participants were asked to run each task for 10 times again without referring to the instruction, assuming they had all become familiar with the tasks during the training. In total, we collected 400 user execution log files from the second session. 20 files were not usable because the user forgot to or did not start the EventLogger correctly, these files were thrown away.

RESULTS

Table 2 lists the result of the user study including the average, maximum, and minimum task execution time and standard deviation. Figure 3 illustrates the predicted and measured execution times for the four methods with prediction error of 2.25%, 1.38%, 7.43%, and 3.88%, correspondingly. The average is 3.7%, which is consistent with the 6% average error rate reported in [5].

[1] All work described in this paper was done on Windows platform.

Method	User time (sec)			Standard Deviation
	Average	Max	Min	
Map Nav.	9.00	11.92	7.31	1.38
Soft KB	12.84	15.65	10.14	1.64
Graffiti	13.60	16.24	11.06	1.80
Scroll Bar	10.30	12.26	8.90	0.95

Table 2 Task execution time from user study

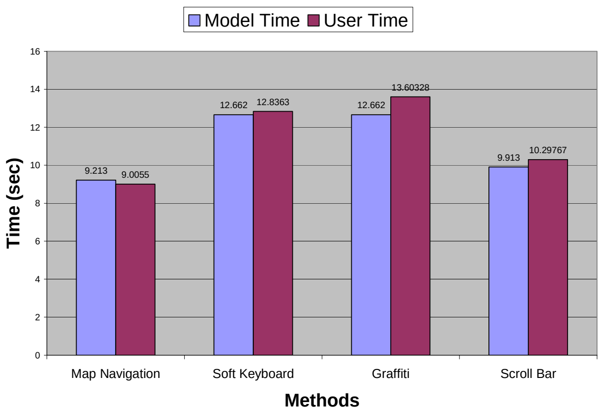


Figure 3 Task execution time: predicted vs. measured

DISCUSSION

The highest prediction error is in the Graffiti task, which might be attributed to individual differences in skill with Graffiti. The longest task time for the Graffiti method came from users 2 and 7, both of whom were pocket PC owners. Since they did not use Graffiti as often as other users, they might need to spend longer thinking time before making the Graffiti strokes. By removing their data from the average user time, we got a lower prediction error of 3.8% for the Graffiti method.

The modeling process and resulting predictions have revealed two important issues that had not been addressed before. First, the stylus-based interface on handheld devices introduces the need to update the original KLM parameters and rules. During the early stage of this study, we found that the Graffiti stroke should be added as a new operator to address the time for the user to make the stroke and for the system to recognize the character. We used 580ms for the Graffiti operators in our study, based on a previous study [3]. Second, the location and value of the R(*t*) operator has become more important to the predicted task time in handhelds than it has been in many of the KLMs in the literature. For the tasks studied, we identified the locations where R(*t*) should be explicitly added to the model, and estimated the corresponding parameter *t*. For example, the estimated system response time to load the museum list was 4200 ms, and the time to update the list was 2300 ms.

CONCLUSION AND FUTURE WORK

We believe this work to be the first attempt to validate the KLM on stylus-based interfaces of handheld devices. We found that the system response time operator R(*t*) should be carefully estimated in an interactive system where the accuracy of model prediction can greatly affected otherwise. We plan to perform a larger set of similar studies on other handheld devices such as pocket PCs and smart cell phones, and add more representative tasks into this study. We also plan to study the KLM on other novel interfaces including speech, gesture, and even eye movement.

ACKNOWLEDGEMENT

This research was supported by funds from the Office of Naval Research, N00014-03-1-0086 and the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHD030010. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Office of Naval Research or the U. S. Government, the DARPA or the Department of Interior-National Business Center (DOI-NBC).

REFERENCES

1. Anderson, J.R. and Lebiere, C. *The Atomic Component of Thought*. Lawrence Erlbaum Associates, Hillsdale, NJ, USA, 1998.
2. Card, S.K., Moran, T.P., and Newell, A. The Keystroke-Level Model for User Performance Time with Interactive Systems. *Communications of the ACM archive*, 23(7), 1980, 396-410.
3. Fleetwood, M.D., Byrne, M. D., et. al. An evaluation of text entry in Palm OS-Graffiti and the Virtual Keyboard. *Proceedings of the Human Factors and Ergonomics Society 46th Annual Meeting.* Santa Monica, CA, 2002.
4. Haunold, P. & Kuhn, W. A keystroke level analysis of a graphics application: Manual map digitizing. In Proceedings of CHI, 1994, Boston, MA, USA, April 24-28, 1994). New York: ACM, 337-343.
5. John, B.E., Prevas, K., Salvucci, D.D., and Koedinger, K. Predictive Human Performance Modeling Made Easy. *Proceedings of ACM CHI'04 Conference on Human Factors in Computing Systems*, 2004.
6. Koedinger, K.R., Aleven, V., and Heffernan, N. Toward a Rapid Development Environment for Cognitive Tutors. *Proceedings of AI-ED 2003*, 455-457
7. Salvucci, D. D., & Lee, F. J. Simple cognitive modeling in a complex cognitive architecture. CHI 2003, ACM Conference on Human Factors in Computing Systems, CHI Letters 5(1), 265-272.

The End