EverTutor: Automatically Creating Interactive Guided Tutorials on Smartphones by User Demonstration

Cheng-Yao Wang, Wei-Chen Chu, Hou-Ren Chen, Chun-Yen Hsu, Mike Y. Chen
Mobile and HCI Research Lab, National Taiwan University
{r00944052,r01922013,r01922002,b99204007,mikechen}@csie.ntu.edu.tw

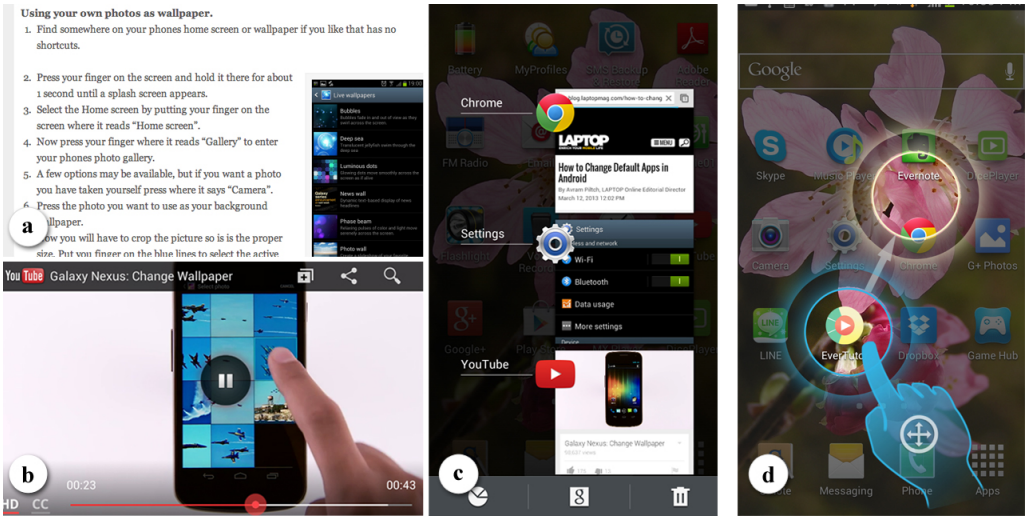

Figure 1. Common types of smartphone tutorials: (a) Static text and images, (b) Video, (c) EverTutor's interactive, step-by-step guided tutorial. Compared to EverTutor which overlays each step on top of the corresponding screen, static and video tutorials require users to constantly switch between the tutorial and the primary task, as shown in (d).

ABSTRACT

We present EverTutor, a system that automatically generates interactive tutorials on smartphone from user demonstration. For tutorial authors, it simplifies the tutorial creation. For tutorial users, it provides contextual step-by-step guidance and avoids the frequent context switching between tutorials and users' primary tasks. In order to generate the tutorials automatically, EverTutor records low-level touch events to detect gestures and identify on-screen targets. When a tutorial is browsed, the system uses vision-based techniques to locate the target regions and overlays the corresponding input prompt contextually. It also identifies the correctness of users' interaction to guide the users step by step. We conducted a 6-person user study for creating tutorials and a 12-person user study for browsing tutorials, and we compared EverTutor's interactive tutorials to static and video ones. Study results show that creating tutorials by EverTutor is simpler and faster than producing static and video tutorials. Also, when using the tutorials, the task completion time for interactive tutorials were 3-6 times faster than static and video tutorials regardless of age group. In terms of user preference, 83% of the users chose interactive type as the preferred tutorial type and rated it easiest to follow and easiest to understand.

Author Keywords

Tutorials; Contextual help; Touchscreen gesture; Smartphone.

ACM Classification Keywords

H.5.2. Information Interfaces and Presentation (e.g. HCI): Graphical User Interfaces.

INTRODUCTION

Smartphones have become ubiquitous in our daily lives. For the times when users need help, they currently resort to finding tutorials that are static or video, and by asking friends to demonstrate how it is done. For static tutorials, text and images are arranged to describe the steps of operations required to accomplish a task. Video tutorials are screen recordings that show the process of performing a task (see Figure 1). Both of static and video tutorials frequently require users to switch between the tutorials and user contexts. In addition,

CHI'14, April 26–May 1, 2014, Toronto, Canada.

http://dx.doi.org/10.1145/2556288.2557407

users often repeatedly pause and replay the videos. This out-of-context process indicates two usability problems: split attention and delayed practice [18].

Contextual interactive tutorials, such as the ones when an Android device is used for the very first time, address these usabilities problems, but they require significant programming skills and privileged access to system source code. Recent research on tutorial content have included contextual associations, such as Stencils-based tutorials [14], Graphstracts [13] and "Photo Manipulation Tutorials" [8]. However, these contextual tutorials are limited to particular applications and tasks and can't be viewed on smartphones. Vision-based analysis are applicable to any GUI regardless whether the source code and accessibility APIs are available [26]. However, users need to write Sikuli scripts [25] and take screenshots for each step; in addition, touchscreen gestures are not supported.

We present EverTutor, a system that automatically generates interactive step-by-step tutorials from user demonstration. EverTutor records low-level touch events, distinguishes different touchscreen gestures and extracts regional screenshot around event location to generate tutorials automatically. When a tutorial is browsed, the system uses vision-based techniques to locate the target interactive region in a given tutorial step. Besides, EverTutor identifies the correctness of the user's interaction and gives response, making the contextual help more interactive. Moreover, EverTutor even automatically injects scroll or swipe event to find the unseen interactive region for users.

We compare the interactive step-by step tutorials generated by EverTutor with static and video tutorials. To evaluate user experience, user preferences, the overall time of creating tutorials, and the task completion time, we conducted a 6-person user study for creating tutorials, and another 12-person user study for browsing tutorials. In the first study, we demonstrated that creating tutorials by EverTutor is simpler and faster than producing static and video tutorials. According to the second study, interactive tutorial is 3-6 times as fast as static and video tutorials. In addition, with interactive tutorials, older users can complete a task as fast or faster than younger users. When participants were asked to rate all the methods in terms of 1) easy to follow, 2) easy to understand, and 3) easy to remember, interactive tutorial is rated highest in easy to follow and easy to understand. In addition, 83% of the users chose interactive type as the preferred tutorial type.

In summary, the main contributions of this paper include:

- a tutorial system that simplifies the generation process of contextual tutorials across apps and supports general touchscreen gestures on Android smartphones.
- a general approach to automatically generating interactive step-by-step tutorials from demonstrations.
- presenting interactive step-by-step tutorials that help older users complete a task as fast as youngsters
- an evaluation of interactive step-by-step tutorials comparing with static and video tutorials on smartphone.

RELATED WORK

Capturing touchscreen events

Gomez et al. described RERAN that permits record-and- replay for the Android smartphone platform by capturing and replaying the low-level events triggered on the phone [6]. Henze et al. derived a compensation function that shifts the users' touches to reduce errors from the large amounts of touch events, which were collected from a published smartphone game in the Android Market [12]. Daryl et al. recorded user touch inputs and improved touch accuracy on mobile devices by learning user-specific touch input models [23]. In EverTutor, the system analyzes the recorded low-level touch events and regional screenshot around touch event location to generate interactive contextual tutorials automatically.

Generation of tutorials by user demonstrations

Most existing tutorial generation approaches trace user interactions by analyzing screencast video of the demonstration or the application. Grabler et al. presented a system that automatically creates tutorials from recorded demonstrations of application usage [8], and Berthouzoz et al. developed a framework, which learned the parameters for image editing operations and generated macros that take into account the context of the current image [3]. Chronicle captures video and history of graphical documents to create an interactive learning tool that offers video playback and visualization of user actions [10]. The MixT tutorial system [4] uses a command log, an input device log and screencapture video to generate mixed media tutorials that combine the strengths of static with video tutorial types. However, tutorials generated by above creation tools must be viewed outside the actual user interface.

Contextual Assistance

Contextual help has been proved effective for learning graphical user interfaces [14, 9, 2]. Unlike traditional help based on screenshots or screencasts, contextual help allows users to receive assistance in the actual interface they are interacting with. Researchers have explored a range of tutorial systems that are presented in the context of the application [14, 9, 5]. Stencils-based tutorials overlay step- by-step instructions on the interface and limit interaction to UI elements related to the current step [14]. Sketch-Sketch Revolution offers in-application tutorials that assist the user in completing drawing tasks, enabling the user to experience success while learning [5]. ToolClips provides users with short video clips and other rich help content next to toolbar buttons contextually [9]. However, these contextual tutorials are limited to specific applications or suited only for a single, specific task.

Yeh et al. developed a tool [26] that allows help designers to create contextual help across applications by taking screenshots and writing Sikuli script language [25]. When the help is presented to a user, the system uses computer vision method to locate the component on user's screen that matches the screenshot. Though we also use vision-based techniques to locate the target interactive region in a given tutorial step, with the recorded low-level touch events, EverTutor distinguishes different touchscreen gestures during the

automatic generation process of tutorials. Also, when a tutorial is presented, EverTutor identifies the correctness of the user's interaction and gives response, making the contextual help more interactive. In addition, EverTutor even automatically injects scroll or swipe event to help the user find the interactive area which is unseen in current screen.

Evaluation of different tutorial types

Different findings on the effectiveness of media formats of software tutorials are shown. The work in the early 90's by Palmiter, Elkerton [19, 20], and Harrison [11] studied the effect of animated demonstrations on learning and instruction recall. More recently, Grabler et al. showed that automatically generated text and image tutorials outperformed video or book instructions on time and errors [13]. ToolClips short (1025 sec.) video demonstrations of features displayed as tooltips has been shown to result in significantly improved task-completion rates as compared to static text+image tooltips [9]. The MixT multimedia tutorial system reported that short, step-specific video demonstrations are useful for performing dynamic actions that are difficult to express via text or static images alone [4]. Pause-and-Play [21] improves a user's ability to follow video demonstrations by automatically pausing and resuming video to stay in sync with the user's progress. Lafreniere et al. designed FollowUs [16] that brought enhanced community features to the tutorial experience, including additional demonstrations of each tutorial step, and noted that the presence of multiple demonstrations helps users complete tasks with less frustration. Differently, we perform an evaluation of EverTutor to understand the user experience of browsing three tutorial types on smartphone and users' preferred type.

DESIGN GOALS

In this section, based on previous research and smartphone characteristics, we outline the design goals of our tutorial system, EverTutor.

Contextual

Although users can long press the home button or use other app-switching apps to switch back and forth between user context and tutorials, this process could cause delayed, disruptive, inconsistent, and obtrusive user experience [1, 15]. Therefore, our first goal is to make our tutorials contextual.

Touchscreen gestures support

Touchscreen is the factor that has changed the way people use mobile phones and tablets. When using smartphones, users perform many touchscreen gestures, such as scrolling through a web page, pinching to zoom in a photo, and dragging an app icon to home screen. As a result, one of our goals is to enable the 7 touchscreen gestures supported by Android to be contained in the tutorial [22].

Interactive

Inspired by Stencils-based tutorial [14] and Google SketchUp Training [7], we intend to design the tutorials that are aware of a user's interactions and able to respond. In addition, since the smartphone screen is too small to show directly the unseen targets in several situations, we expect EverTutor to automatically find the unseen target area for users, making our contextual help more interactive and improving users' learning experience.

Automatic generation from user demonstration

In order to simplify the generation process of contextual tutorials, we intend to allow users to create tutorials during their demonstration. Considering smartphones having so many apps and features, automatic generation could possibly meet smartphone users' increasing demand for tutorials.

Across apps and platforms

We expect our tutorial system to be suited for any task or any app on smartphone and hope our approach to be easily applied to any other touchscreen-based devices.

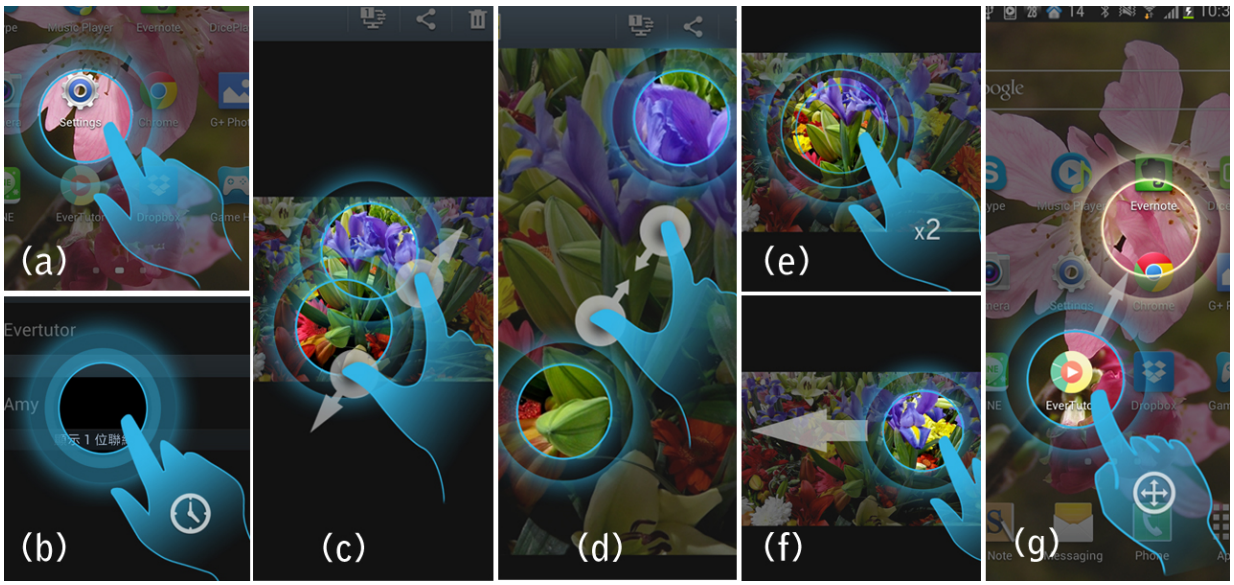

Figure 2. The overlays for touchscreen gestures in EverTutor. (a) Touch (b) Long press (c, d) Pinch open/close (e) Double touch (f) Swipe (g) Drag.

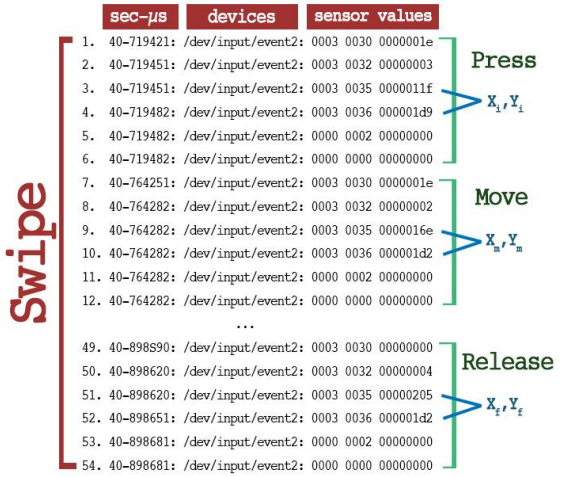

Figure 3. Series of events representing a swipe; X_i, Y_i are the initial coordinates, X_m, Y_m are the coordinates during moving (still holding down), and X_f , Y_f are the final coordinates.

SYSTEM IMPLEMENTATION

We developed EverTutor on a Samsung Note 2 smartphone with root access running Android 4.1.2.

Gesture Recognizer

The Android software stack consists of a custom Linux kernel and libraries, the Dalvik Virtual Machine (VM) and apps running on top of the VM. When a user interact with an Android app, the Android device's sensors generate and send events to the kernel via the /dev/input/event* device files. The standard, five-field format of every event is as follows: timestamp:, device:, type, code, value, and touchscreen gestures are encoded as a stream of touchscreen events in the same format, which could be displayed as the following example:

40-719451: /dev/input/event2: 0003 0035 0000011f.

The timestamp indicates that the event was generated 40 seconds and 719,451 microseconds since the system restart; "event2" is the input device which refers to the touchscreen on our device. The next three columns provide position information: 0035 represents the x position of the event and 0000011f (hex) corresponds to coordinate 287 (decimal) of the screen. However, this single event is not enough for reconstituting the high-level gesture. For instance, a single press usually involves roughly 18 touchscreen events, while a swipe usually involves roughly 50 touchscreen events. In Figure 3, we show a subset of events involved in a single typical gesture, swipe. We implemented a gesture recognizer to distinguish different gestures based on the default properties of a gesture defined by Android, for example, standard long press time is 500 ms.

Template Matching

We implemented template matching based on image pyramid (3 levels), where the results of each layer is used to define the region of interest for the next layer. To improve performance, we implemented it in C++ using the OpenCV library and used Android's Native Development Kit (NDK) to access the function via Java Native Interface (JNI).

Client-server Architecture

EverTutor is constructed under a client-server architecture. When a tutorial is created by EverTutor, all images are uploaded to Amazon S3 and a tutorial data is newed in the Database of Rails Server. When a user browses a tutorial, EverTutor downloads all necessary images from Amazon and accesses the step-by- step tutorial data from Rails Database, such as event type, touchscreen position, etc..

EVERTUTOR

Tutorial Format

In order to free users from switching back and forth between a tutorial and user context, we design the interactive and contextual guide that shows step-by-step instructions on the actual interface rather than in a separate viewer. Inspired by the "Cling" view from Launcher on Android Jelly Bean and Stencils-Based Tutorials [14], we put full-screen and transparent overlays containing cling holes upon the active application interface. In every step, the cling holes in the overlay layer draw the user's attention to the region they should interact with; Also, we add a finger image beside the cling hole to present the 7 core gestures supported by Android as shown in Figure 2. Hoping to direct users correctly, users can proceed to the next step only after interacting with the intended position by the expected gesture.

Generating tutorial with EverTutor

Record and distinguish touchscreen gesture

When creating tutorials with EverTutor, the user starts recording by performing a swipe gesture to select the start item in pie menu. System keeps reading the /dev/input/event* files, detecting the timestamp, position and event types to distinguish the used gesture. Meanwhile, system opens the framebuffer (/dev/graphics/fb0) to capture bitmap data representing the screenshot and extract the regional image around the touch event location (Figure 4). An Android toast message pops up when a tutorial step data is saved completely. Besides tutorial authors can perform mode switching by using a bezel swipe gesture to open a pie menu containing the mode choices.

Record Options

During the recording process, gestures other than scroll and swipe (e.g. tap, pinch) are regarded as separate steps, and we exclude scroll and swipe because the two gestures are possibly applied by the users for many times to find a specific target. On the other hand, when a tutorial is browsed, gestures of swipe and scroll recorded can be analyzed by EverTutor to find the interactive area unseen in the current screen for users.

Scroll/swipe mode

Some swipe and scroll gestures are used to extend the hidden interface; for instance, in Google Maps, place details, such as business information, can be accessed by swiping up on the

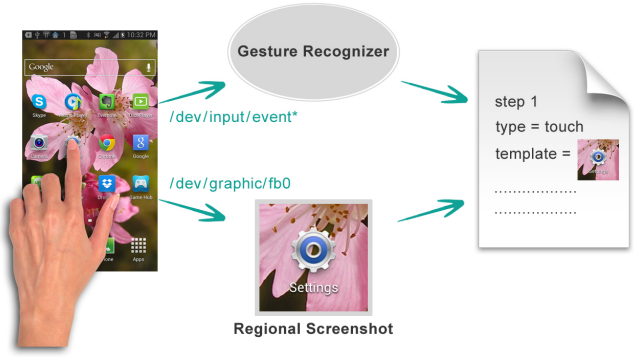

Figure 4. The workflow of generating tutorial with EverTutor.

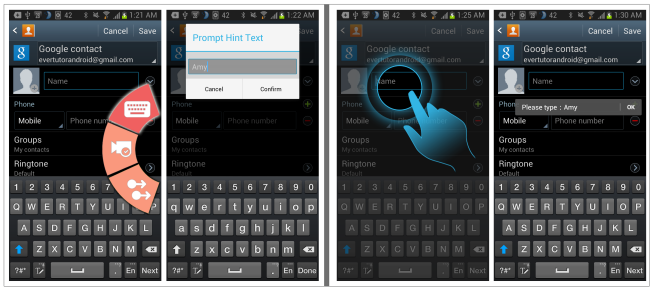

Figure 5. (Left) Switch to typing mode and set the prompt instructional text in dialog. (Right) The user inputs the texts by following the prompt instruction.

info sheet header. In the above situations, the tutorial author can switch to "scroll/swipe" mode to include the two gestures into dependent steps.

Typing mode

For steps containing typing, the user can select "typing" mode to stop capturing the sensitive information, such as username and password, and after finishing typing, the user can set the prompt text in pop up dialog for users browsing the tutorial afterward as shown in Figure 5(left).

Set title and review tutorial

After completing all steps, the tutorial author swipes to open the pie menu selecting "finish" icon, titles the tutorial and checks the screenshots for every step, and then the tutorial is finally inserted into database with all images uploaded to Amazon S3.

Browsing tutorials with EverTutor

EverTutor firstly downloads interactive region images of every tutorial step from Amazon S3 and the tutorial information from database, such as event type and event location of every step.

Finding the interactive region

When a user browses a tutorial, EverTutor continuously reads the screen framebuffer (/dev/graphics/fb0) and scans for regions that match the templates. Currently, it takes 0.1 0.3s to match a 240x240 pixel template image within a 1280x720 pixel image on a Samsung Note 2. If the matching is failed and the recorded operation consists of swipe or scroll gesture,

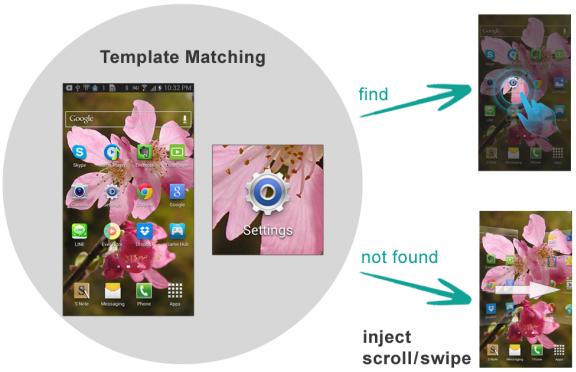

Figure 6. The workflow of browsing tutorials with EverTutor.

system would inject swipe or scroll events to find the interactive area (Figure 6). For example, in the tutorial of "changing system font size to large," the system automatically scrolled up the list in "Settings" to find "Accessibility." If no templates are matched after auto-scrolling, the system will show the template image and prompt users to find the target.

Showing the interactive step-by-step tutorial

After locating the interactive region, based on the type of touchscreen gestures in the given step, the system puts a full-screen and transparent overlay containing cling holes and a finger image upon the active application interface. When a typing action is asked in the step, the user can input the texts following the prompt hint as shown in Figure 5(right).

Identifying the correctness of the user's interaction

Aiming to direct users correctly, the system reads files in /dev/input/event* to obtain touch locations and identify the performed touchscreen gestures. Therefore, users can proceed to the next step only after interacting with the intended position by the expected gesture.

EVALUATION

Task and Tutorial Materials

We selected 9 smartphone tutorials of different tasks from courses held by Samsung and HTC as well as common questions on the telecommunications company websites. The tutorials were classified into three levels by the number of steps required to finish the task, including easy (4 steps), normal (8 steps) and hard (16 steps) as shown in Table 1. Because the original tutorials may not consist of the exact steps of the three predefined levels, we slightly modified them by adding or deleting a few steps. "Typing" steps were included in the hard level with typing characters limited to exactly 13 in each tutorial, and all characters were on the default keyboard. Touchscreen gestures contained by the 9 tutorials were as follows: touch, pinch, swipe, scroll and long press. We didn't present "drag" and "double touch" in our tutorials because drag (consisting of long press, move and lift) is more difficult and double touch is seldom used in smartphone operations.

At each level, 3 different tasks were designed; for each task, we created 3 types of tutorials, static, video, and interactive type (EverTutor), and made these 3 types present as equivalent the information as possible. Our video tutorials were

	Easy (4 steps)	Normal (8 steps)	Hard (16 steps containing entering 13 characters)
T1	Change system font size to large.	Check the battery level and adjust the screen brightness.	Create a new contact (name: Amy, phone number: 0923456789) and practice long press gesture to share.
T2	Enable WiFi hotspot to share the internet.	Set a repeating alarm on Tuesday and Thursday.	Use Evernote to add a note (enter "Evertutor hi.") including an image and a time reminder.
T3	Install Facebook app from Google play.	View photos with pinch and swipe gesture and add a photo to favorite.	Send an email to a@a.a (subject:hi, content:hihihi)

Table 1. The nine tasks users were asked to perform in user study, arranged by level.

recorded by Screencast Recorder app, with subtitles and red circles added to direct users. Though the default delay between each step was 2 seconds, participants were asked to view a video tutorial and optional adjust the playback speed during the practice session prior to the study tasks. Only 2 (older) participants slowed the speed down to 0.8x. Besides, we provided no vocal explanation in video tutorials to avoid situations when users only relied on auditory instructions rather than screen content. To generate our static tutorials, we extracted the frame of each step in the video tutorial and added texts for explanation. For interactive tutorials, we selected the tutorials that were created by users using EverTutor in user study 1.

User Study 1: Creating Tutorials

The goal of the study is to explore if the 3 tutorials of the same level have equivalent difficulty and understand the user experience of generating interactive tutorials by EverTutor as well as users' preferred types to create tutorials.

Participants

We recruited 6 app developers (4 male, 2 female), ranging in age from 20 to 26.

Procedure

We firstly familiarized participants with EverTutor operation, and then asked them to use EverTutor to create 9 designated interactive tutorials ordered by the level from easy to hard. Before making each tutorial, participants must understand every step of the task, and after performing it once, participants used EverTutor to complete the task and generate interactive tutorial simultaneously. After 9 tutorials were all created, we showed participants the generated tutorials and conducted a 15-minute interview. During the study, we used a camera to record user performance.

Quantitative Result

We calculated the difference in operation time for the 3 tutorials of the same level: the difference is less than 3s in 4-step level, 5s in 8-step and 7s in 16-step. The 3 tutorials not only consist of the same number of steps but also cost similar operation time, which proves that they can be grouped into the same difficulty level.

According to Figure 7, the average processing time of each step cost by EverTutor is 2.4s, and plus the time spent on typing the tutorial title and checking for correctness, the overall time is respectively 41.1s, 60.47s, and 128.87s for easy-level, normal-level, and hard-level task.

In Participants' estimate (see Figure 9), they would spend about 20 minutes on making a normal-level tutorial in static type and 40 minutes on a video tutorial of the same level, which obviously shows that creating tutorials by EverTutor is simpler and faster.

Qualitative Result

According to the questionnaire result as shown in Figure 9, all the participants preferred EverTutor to make interactive tutorials. Participant D6 commented, "As I demonstrate a smartphone feature to my parents, the tutorial is generated automatically by EverTutor, and they can browse the tutorial whenever they wants." However, four participants would like to make static or video tutorials as well. "Static tutorial could have more detailed instructions for each step and is suitable

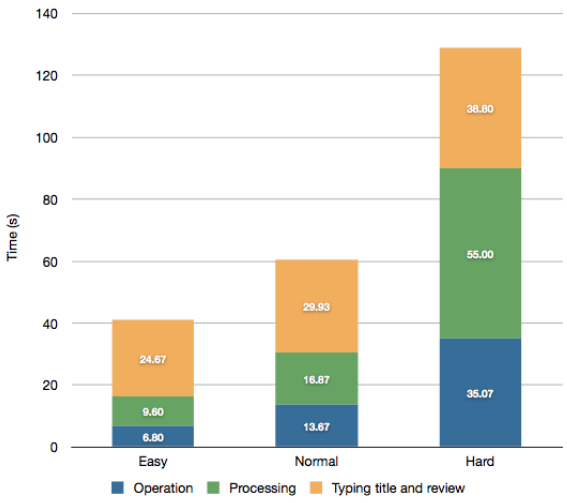

Figure 7. Operation time and overall time for creating tutorials with EverTutor.

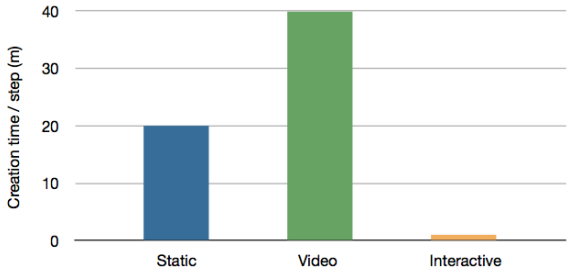

Figure 8. Overall time for creating tutorials.

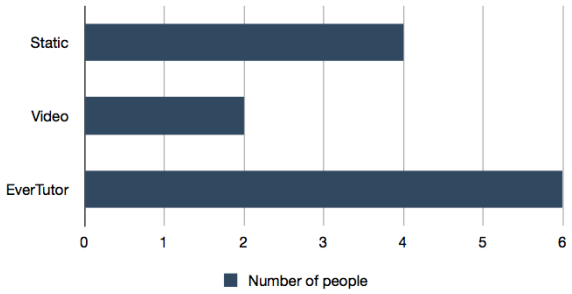

Figure 9. The tutorial types that tutorial authors prefer to create (multiple-selection question).

for the browse on the computer," said D1. D2 said that additional static and video tutorials could offer users more access to the tutorial.

User Study 2: Browsing Tutorials

The goal of this study is to understand the user experience of three types of tutorials (static, video, interactive) and users' preferred types.

Participants

12 Android Smartphone Beginners were recruited, including 6 older participants (3 female, 3 male) ranging in age from 48 to 58 (median 54.3) and 6 young participants (4 female, 2 male) ranging in age from 21 to 29 (median 24.2).

Procedure

We first familiarized participants with basic smartphone gestures and demonstrated three types of tutorials. In static type, using "Chrom" app as the browser, we informed users about the pinch gesture to zoom in the tutorial website and scroll gesture to roam the steps. For video type, we provided "DicePlayer" app, which can adjust the speed of video and taught them the essential operations to manipulate the movie player, including pausing, playing, resuming, forwarding, rewinding, and replaying. Also, participants learned how to switch between user context and the tutorial with home button. In interactive type, they ran through a completely interactive tutorial and got familiar with the indicated gestures. In the experiment, we prepared 3 home screens in launcher to display 41 app icons, referring to the average number of apps per smartphone [17]. In each task of static and video types, user must swipe across home screens to find the target app, but with EverTutor, the target was automatically pointed out.

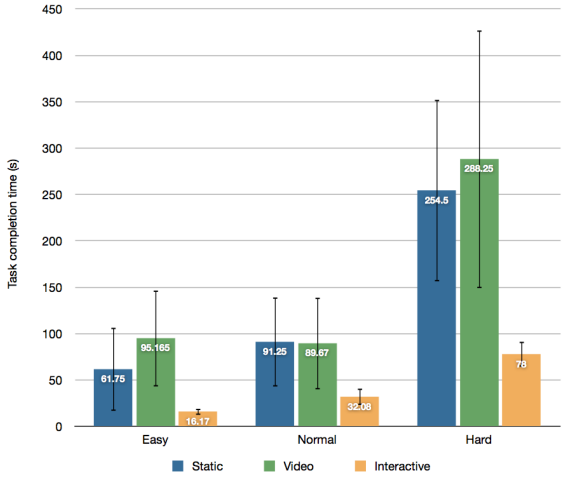

Figure 10. Task completion time with static, video and interactive tutorial.

Afterward, participants were asked to complete nine tasks with tutorials in the order of the difficulty level from easy to hard, and the sequence of tutorial types in each difficulty is counterbalanced. After the 9 tasks, participants filled out a questionnaire and had a 15-minute interview. During the experiment, we used a camera to record user performance.

Quantitative Result

Figure 10 shows the average completion time of 3 tutorial types in each level. There is a significant difference across three tutorial types in each level ($F_{easy}(2,33) = 10.09$, $P_{easy} < .001$, $F_{normal}(2,33) = 11.31$, $P_{normal} < .001$ and $F_{hard}(2,33) = 16.76$, $P_{hard} < .001$). Interactive tutorial is fastest and 3 times as fast as static and video tutorials in all levels .($p < 0.05$). On the other hands, static tutorial is slightly faster than video tutorial in "easy" and "hard" levels, yet there is no significant difference ($p > 0.05$). Besides, it tooks twice the time for older participants to complete the task with static and video tutorials than young participants. Surprisingly, with interactive tutorial, the completion time of older participants is equal to or even less than young participants' as shown in Figure 11.

Qualitative Result

Figure 12 shows the subjective ratings for all tutorial types regarding 1) Easy to Follow, 2) Easy to Understand, and 3) Easy to Remember. Participants rated interactive tutorial the highest in all categories, and a significant difference is shown in Easy to Follow and Easy to Understand ($F_{follow}(2,33) = 20.23$, $P_{follow} < .001$ and $F_{understand}(2,33) = 3.78$, $P_{uderstand} < .05$). One participant stated, "The interactive tutorial is intuitive and easily understood. I don't need a frequent switch between actual interface and the tutorial." Another participant reported, "Interactive tutorial is like a real person who teaches me step by step, and it can find the app for me automatically. It is really cool."

When asked about most preferred tutorial type, 83% of the users chose interactive type over the other types, as shown in Figure 13. The reason why the two users preferred static type

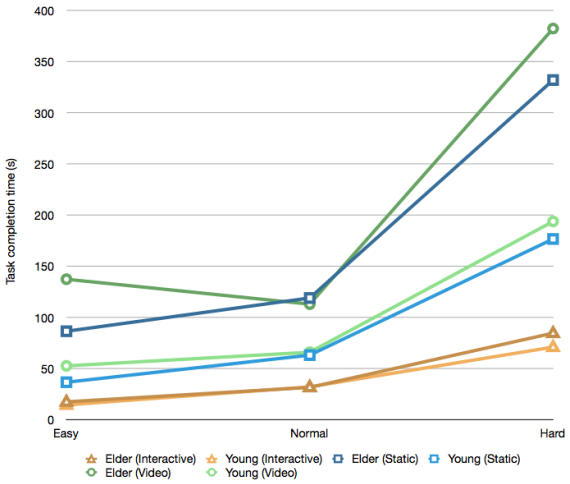

Figure 11. Task completion time of older and young participants with all tutorial types.

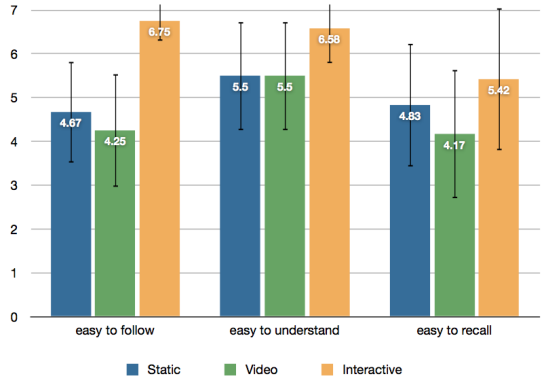

Figure 12. Subjective results comparing the interactive tutorials with static and video tutorials (7-Likert scale).

is that they could realize the relations among steps and jump to the critical step directly. Besides, the step-by-step interactive type is too intuitive for users to remember the whole tutorial.

DISCUSSION

Significant gain in performance

EverTutor's gain in performance is significantly greater than observed in Stencils[14] (3x faster vs 26% faster completion time compared to static tutorials). It's not only because context switching is much more difficult on smartphones, but also because EverTutor's interactive tutorial is significantly easier for users to understand and follow.

Differences between older and younger participants

Older participants had equal or faster completion time using interactive tutorials compared to younger participants, but spent at least 2 times longer than young participants when using static and video tutorials, which indicates that contextual tutorials are particularly well-suited for older users. One of our initial concerns with the interactive step-by-step tutorials

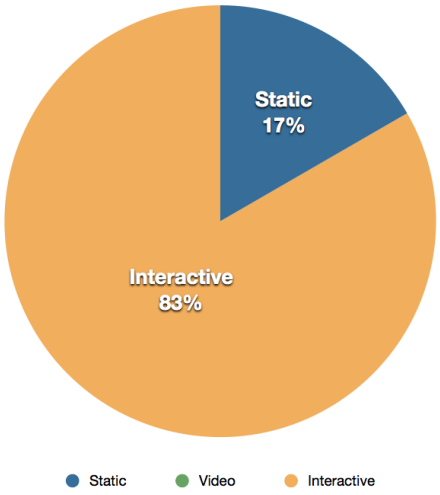

Figure 13. Users' preferred tutorial types.

was that users might follow tutorials quickly without remembering what they were learning. According to our questionnaire result in Figure 14, older participants rated interactive tutorials easiest to remember (6.0 vs 4.2 vs 3.8 on a 7-point scale) whereas younger participants rated static tutorial easiest to remember (5.5 vs 4.8 vs 4.5).

An older participant said, "Despite the detailed explanation of static tutorial, it's hard for me to remember the whole steps. In contrast, by using interactive tutorials, I can just memorize the interactive regions." According to another older user, though he couldn't memorize all the steps after following the interactive tutorial by one time, he could learn fast by repeating the tutorial for some more times, which was much easier than watching static or video types for several times. However, two younger participants preferred static type. A young user considered, "the relation among steps is clear in static tutorial, and I can understand the tutorial more; but I tend to forget what I've done after following interactive tutorials."

Automatic scrolling feature

10 of the 12 participants regarded the function of finding the unseen target region automatically very useful especially when users hoped to find an app icon on Launcher, but others commented that they wanted the system to just show the regional screenshot of target region and found the target by themselves.

Automation macro

During the interview, users suggested that EverTutor could generate a shortcut that automatically completed the task for users. However, since hands-on practice and exploration at the same time have been shown to be key ingredients of successful learning of an interface [24], the shortcut feature would be suited only for the simple functions that users had already learned.

LIMITATION AND FUTURE WORK

The current implementation of EverTutor has some limitations that should be discussed. EverTutor can't generate

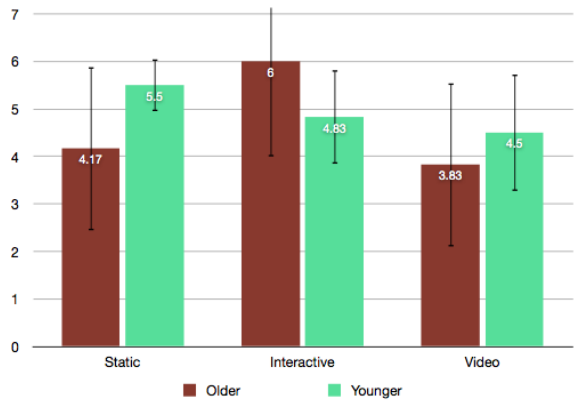

Figure 14. Subjective results in easy to remember catagory of older and younger participants.

tutorials containing motion gestures like throwing and continuous, complex operation, such as drawing and playing a game. A possible solution worth pursuing in the future is to record the whole low-level events and replay them when user browses the step. With the record-and-replay approach, EverTutor can also automatically make the user jump to a particular step by replaying the recorded events of previous steps.

Besides, our system, like any other vision-based techniques, suffers from the limitations like scale changes and theme variations. To alleviate the problem, EverTutor can record the information of system version, device name, and some details about the apps, such as package and activity names in each tutorial step, to assist users in finding tutorials generated from similar system environments. Users also mentioned that they preferred to add some instructional texts when making tutorial steps, which will be addressed in our future work.

CONCLUSION

In this paper, we present EverTutor that automatically generate interactive step-by-step tutorials on smartphone. EverTutor analyzes the recorded low-level touch events, distinguishes different touchscreen gestures and extracts regional screenshot around touch event location to generate step-by-step tutorials automatically. The system locates the target interactive regions of each step to overlay tutorial contextually. To make the tutorial more interactive, EverTutor is aware of the user's interaction and gives response. Besides, EverTutor automatically scrolls or swipes to help users find the unseen interactive region.

To compare interactive step-by step tutorials with static and video tutorials, we firstly conducted a 6-person user study for creating tutorials. The result shows that creating tutorials by EverTutor is simpler and faster than producing static and video ones. In the second 12-person user study for browsing tutorials, we demonstrated that interactive tutorial is fastest in task completion time and 3 times faster than static and video tutorials. Moreover, interactive tutorials even enable older users to complete a task faster than youngsters. To rate all the methods in terms of 1) easy to follow, 2) easy to understand and 3) easy to remember, interactive tutorial was highest rated in easy to follow and easy to understand. In addition, 83% of the users chose interactive type as the most preferred tutorial type.

ACKNOWLEDGMENTS

We gratefully acknowledge the helpful comments of the Associate Chairs and the anonymous reviewers. This paper was partially supported by National Science Council of Taiwan under 101-2628-E-002-030-MY2.

REFERENCES

1. Ames, A. L. Just what they need, just when they need it: An introduction to embedded assistance. In *Proc. ACM SIGDOC'01 (2001)*, 111–115.
2. Bergman, L., Castelli, V., Lau, T., and Oblinger, D. Docwizards: A system for authoring follow-me documentation wizards. In *Proc. ACM UIST'05 (2005)*, 191–200.
3. Berthouzoz, F., Li, W., Dontcheva, M., and Agrawala, M. A framework for content-adaptive photo manipulation macros: Application to face, landscape, and global manipulations. *ACM Trans. Graph (2011)*, 120:1–120:14.
4. Chi, P.-Y., Ahn, S., Ren, A., Dontcheva, M., Li, W., and Hartmann, B. Mixt: Automatic generation of step-by-step mixed media tutorials. In *Proc. ACM UIST'12 (2012)*, 93–102.
5. Fernquist, J., Grossman, T., and Fitzmaurice, G. Sketch-sketch revolution: An engaging tutorial system for guided sketching and application learning. In *Proc. ACM UIST'11 (2011)*, 373–382.
6. Gomez, L., Neamtiu, I., Azim, T., and Millstein, T. Reran: Timing- and touch-sensitive record and replay for android. In *Proc. ACM ICSE'13 (2013)*, 72–81.
7. Google SketchUp Training. `http://sketchup.google.com/intl/en/training/index.html`.
8. Grabler, F., Agrawala, M., Li, W., Dontcheva, M., and Igarashi, T. Generating photo manipulation tutorials by demonstration. In *Proc. ACM SIGGRAPH'09 (2009)*, 66:1–66:9.
9. Grossman, T., and Fitzmaurice, G. Toolclips: An investigation of contextual video assistance for functionality understanding. In *Proc. ACM CHI'10 (2010)*, 1515–1524.
10. Grossman, T., Matejka, J., and Fitzmaurice, G. Chronicle: Capture, exploration, and playback of document workflow histories. In *Proc. ACM UIST'10 (2010)*, 143–152.
11. Harrison, S. M. A comparison of still, animated, or nonillustrated on-line help with written or spoken instructions in a graphical user interface. In *Proc. ACM CHI'95 (1995)*, 82–89.
12. Henze, N., Rukzio, E., and Boll, S. 100,000,000 taps: Analysis and improvement of touch performance in the large. In *Proc. ACM MobileHCI'11 (2011)*, 133–142.

13. Huang, J., and Twidale, M. B. Graphstract: Minimal graphical help for computers. In *Proc. ACM UIST'07 (2007)*, 203–212.

14. Kelleher, C., and Pausch, R. Stencils-based tutorials: Design and evaluation. In *Proc. ACM CHI'05 (2005)*, 541–550.

15. Knabe, K. Apple guide: A case study in user-aided design of online help. In *Proc. ACM CHI'95 (1995)*, 286–287.

16. Lafreniere, B., Grossman, T., and Fitzmaurice, G. Community enhanced tutorials: Improving tutorials with multiple demonstrations. In *Proc. ACM CHI'13 (2013)*, 1779–1788.

17. Nielsen: Average Number of Apps per Smartphone. `http://www.nielsen.com/us/en/newswire.html`.

18. Palaigeorgiou, G., and Despotakis, T. Known and unknown weaknesses in software animated demonstrations (screen-casts): A study in self-paced learning settings.

19. Palmiter, S., E. J., and Baggett, P. Animated demonstrations vs written instructions for learning procedural tasks: a preliminary investigation. 687701.

20. Palmiter, S., and Elkerton, J. Animated demonstrations for learning procedural computer-based tasks. 193216.

21. Pongnumkul, S., Dontcheva, M., Li, W., Wang, J., Bourdev, L., Avidan, S., and Cohen, M. F. Pause-and-play: Automatically linking screencast video tutorials with applications. In *Proc. ACM UIST'11 (2011)*, 135–144.

22. Android core gesture set. `http://developer.android.com/design/patterns/gestures.html`.

23. Weir, D., Rogers, S., Murray-Smith, R., and Löchtefeld, M. A user-specific machine learning approach for improving touch accuracy on mobile devices. In *Proc. ACM UIST'12 (2012)*, 465–476.

24. Wiedenbeck, S., and Zila, P. L. Hands-on practice in learning to use software: A comparison of exercise, exploration, and combined formats. 169–196.

25. Yeh, T., Chang, T.-H., and Miller, R. C. Sikuli: Using gui screenshots for search and automation. In *Proc. ACM UIST'09 (2009)*, 183–192.

26. Yeh, T., Chang, T.-H., Xie, B., Walsh, G., Watkins, I., Wongsuphasawat, K., Huang, M., Davis, L. S., and Bederson, B. B. Creating contextual help for guis using screenshots. In *Proc. ACM UIST'11 (2011)*, 145–154.