TapShot: Screenshot Snippets as GUI Shortcuts

Kristian Gohlke,[*] Michael Hlatky
Hochschule Bremen (University of Applied Sciences)

Jörn Loviscach[†]
Fachhochschule Bielefeld (University of Applied Sciences)

1 Introduction

During extended sessions with a graphical user interface (GUI), users often apply a small set of commands with high frequency. A majority of direct manipulation tasks on a GUI are carried out using the mouse, particularly when keyboard shortcuts are not provided or the user is not familiar with them. Thus, to invoke a certain command, the user is required to aim the mouse pointer at a given on-screen widget and click with the mouse. If the overall task requires a user to click on the same widget repeatedly as part of a sequence of different interleaved micro-tasks, the overall performance suffers, as each point-and-click action requires a considerable amount of time for correctly aiming at the respective control.

This work demonstrates a system that enables quick and transparent access of frequently used functionality in virtually any GUI-enabled legacy application. For this purpose, we employ the capabilities of readily available touch-sensitive devices—such as the iPhone or similar multitouch controllers—to replace repetitive point-and-click mouse interactions. The use of mouse interaction still prevails for precise direct manipulation and for triggering more rarely used commands on a GUI. Carrying over the idea of using GUI screenshots for scripting automated actions [Yeh et al. 2009], we use small screenshots as pushbuttons on the touch device. Controlling software via dynamically labelled keys is also in the spirit of the steeply priced Optimus Maximum computer keyboard (http://www.artlebedev.com/everything/optimus/).

2 System Design

The TapShot system allows quickly assigning functionality from a GUI to a touch screen device through custom visual shortcuts. Each shortcut is represented by a small icon on the touch screen. Initially, each icon's place is blank. When the user interacts with any application through the mouse on the main computer display, shortcuts can be created. This is achieved by pressing any of the icons while simultaneously holding down an additional 'snip' button on the touch display. This invokes a learn mode, which is further indicated by an icon-sized translucent square underneath the mouse pointer on the main screen. The size of the square can be adjusted by spinning the scroll wheel.

Once the user clicks a mouse button, the system grabs a screenshot of the square area and presents the captured snippet on the touch display, scaled to fit into the grid of icons, see Figure 1. Now, touching the icon initiates a mouse click of the previously entered type (left/middle/right, double) at the original screen location, thus providing a shortcut to the underlying command.

This approach does not require any additional information from GUI widgets or the operating system. Thus, it works with any application that reacts to mouse input. As a fast and transparent way of mapping shortcuts to an input device placed conveniently at the fingertips of the user, the system can reduce strain and supports a more efficient interaction when carrying out tasks that require repetitive clicking on the same UI widget. The current version of TapShot employs a client-server architecture that enables its use on virtually any networked touch-screen device.

[*]e-mail: kgohlke@acm.org
[†]e-mail: joern.loviscach@fh-bielefeld.de

SIGGRAPH 2010, Los Angeles, California, July 25 – 29, 2010.
ISBN 978-1-4503-0210-4/10/0007

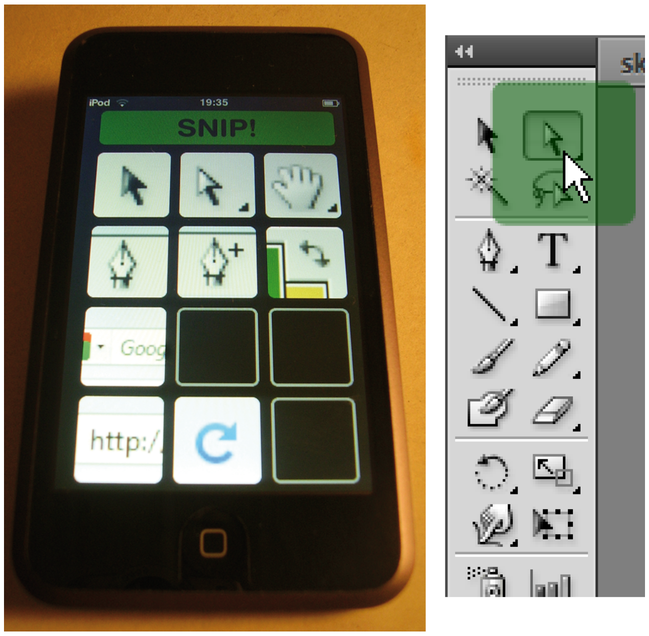

Figure 1: *The TapShot icons collected by the user (left) can be used to invoke commands on the main screen (right).*

3 Outlook

Ongoing work aims at providing a more robust mapping between the selected locations of action and the current state of the GUI of an unmodified legacy application, similar to Besacier et al. [2009] or Myers et al. [2001]. An initial step will be to store location data relative to the coordinates of the window that was active at the time of grabbing screenshots and reposition the mouse click event from the touch screen in case the underlying window's position has changed since the assignment. As an alternative, image processing could be applied to track UI widgets on a completely graphical basis, following the approach of Yeh at al. [2009]. The TapShot system could provide a way to quickly record and trigger sequences of click events or keystrokes on the fly, similar to the macro functionality already present in many applications—with the added benefit of a central and more transparent interface. To control actions that are not directly accessible from a GUI, the TapShot system could further be used to fire MIDI, OSC or TUIO data events.

References

Besacier, G., and Vernier, F. 2009. Toward user interface virtualization: legacy applications and innovative interaction systems. In *Proc. of EICS '09*, 157–166.

Myers, B. A., Peck, C. H., Nichols, J., Kong, D., and Miller, R. 2001. Interacting at a distance using semantic snarfing. In *Proc. of UbiComp '01*, 305–314.

Yeh, T., Chang, T.-H., and Miller, R. C. 2009. Sikuli: using GUI screenshots for search and automation. In *Proc. of UIST '09*, 183–192.