Vocal Shortcuts for Creative Experts

Yea-Seul Kim*
University of Washington
Seattle, Washington
yeaseul1@uw.edu

Mira Dontcheva
Adobe Research
Seattle, Washington
mirad@adobe.com

Eytan Adar*
University of Michigan
Ann Arbor, Michigan
eadar@umich.edu

Jessica Hullman
Northwestern University
Evanston, Illinois
jhullman@northwestern.edu

ABSTRACT

Vocal shortcuts, short spoken phrases to control interfaces, have the potential to reduce cognitive and physical costs of interactions. They may benefit expert users of creative applications (e.g., designers, illustrators) by helping them maintain creative focus. To aid the design of vocal shortcuts and gather use cases and design guidelines for speech interaction, we interviewed ten creative experts. Based on our findings, we built VoiceCuts, a prototype implementation of vocal shortcuts in the context of an existing creative application. In contrast to other speech interfaces, VoiceCuts targets experts' unique needs by handling short and partial commands and leverages document model and application context to disambiguate user utterances. We report on the viability and limitations of our approach based on feedback from creative experts.

CCS CONCEPTS

• **Interaction paradigms** → **Natural language interfaces**.

KEYWORDS

speech interaction, creative applications, expert users

ACM Reference Format:
Yea-Seul Kim, Mira Dontcheva, Eytan Adar, and Jessica Hullman. 2019. Vocal Shortcuts for Creative Experts. In *CHI Conference on Human Factors in Computing Systems Proceedings (CHI 2019), May 4–9, 2019, Glasgow, Scotland UK*. ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3290605.3300562

*Also with Adobe Research.

CHI 2019, May 4–9, 2019, Glasgow, Scotland UK

ACM ISBN 978-1-4503-5970-2/19/05…$15.00
https://doi.org/10.1145/3290605.3300562

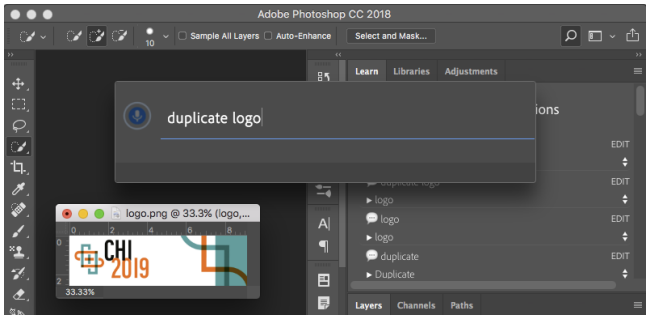

Figure 1: The interface of VoiceCuts includes a panel that opens when the user wants to talk (here with command "duplicate logo") and a customization and history panel that shows recent commands and allows the user to specify custom vocal shortcuts.

1 INTRODUCTION

Professional creative applications for design, drawing, photo editing, or even slide creation support an extraordinarily broad set of features. To maintain creative focus, experts develop or learn practices that accelerate access to common functions. Combinations of procedural memory (i.e., muscle memory) and affordances for accelerating performance, such as keyboard shortcuts, can help to reduce cognitive and physical interaction costs. However, there are many barriers to learning and adopting these behaviors. Keyboard shortcuts, custom panels, and macros can make access more efficient, but past research shows that keyboard shortcuts are hard to learn [20]. Manual interface customization is rarely done by users [38] and automated adaptive interfaces that dynamically change the interface may interfere with the memory for certain widgets that experts develop based on placement [15]. Even those who use an application daily can struggle to find tools they don't use often and may be slow to open the right panel at the right time due to the sheer number of options. Taken together, the *cognitive costs* of learning, recalling, and integrating 'accelerators' into practice may lead even experts to 'settle' for slower interactions. Furthermore, many traditional UIs, and accelerators, assume mouse and keyboard input. For the many creative experts who use tablets and stylus pens as primary input devices, this introduces significant *physical costs* as they must maneuver away from their primary devices to invoke most shortcuts.

We propose that an alternative–*vocal shortcuts*–can better provide access to features for expert users. Vocal shortcuts are short spoken phrases that can be engaged by the end-user *on top* of their existing interfaces. The use of speech introduces a number of benefits. Physically, the end-user does not need to switch from their tablet to a mouse or keyboard. Cognitively, short phrases may be easier to learn and recall than the vaguely mnemonic keyboard shortcuts. Furthermore, because the vocabulary for speech can be broader than for keyboard counterparts, more complex features can be accessed (e.g., those that require parameters) or multi-step workflows may be invoked more easily.

While vocal shortcuts are a promising way to allow creative experts to maintain focus on their work, there are many open-questions for their design. For example, using vocal shortcuts requires learning to integrate a new modality and 'language' into one's existing practice. On the one hand, expert users may already have a rich vocabulary that is closely aligned with system features (i.e., they know features names). On the other hand, speech interfaces have limited discoverability features [11, 16], making grammar and advanced features harder to learn. Additionally, speech-to-text systems may not work as expected (i.e., perfectly). Past experience by end-users with systems such as Apple's Siri or Amazon's Echo may further shape expectations of speech-to-text performance and behavior. Thus, it is not entirely obvious where, if, and how, vocal shortcuts are best applied. Further, unlike novice users, experts have existing optimized workflows and habits to complete tasks [22, 35, 60]. Unless speech input offers significant reductions in cognitive or physical costs, end-users are likely to stick to non-speech modalities [3].

To understand where speech interaction might be helpful in existing creative applications, we interviewed ten creative experts. Though individual examples varied, a number of patterns emerged both for frequent and infrequently accessed features. One common request was for easier ways to access things that had 'names.' Names are often assigned by the end-user but lack an application shortcut. Examples include layers (e.g., the 'mouth layer'), brushes (e.g., 'three pixel butterfly brush'), or colors (e.g., 'goldenrod'). Because speech input was viewed as a way of bypassing complex menus, a number of participants identified vocal shortcuts as an effective way to access infrequently used features.

Based on these interviews, we built VoiceCuts, an extension to Adobe Photoshop to support speech as an input modality. The system responds to *short* commands for selecting tools and menus, changing parameters, and manipulating layers of the document. VoiceCuts also provides adaptability features to support custom vocal shortcuts. To evaluate the viability of vocal shortcuts, we invited eight experts to do a creative task in our lab and deployed the system with one creative expert to use in his own environment. We identify situations where speech can be useful for expert users and limitations of the approach. Participants found our speech-enabled prototype helpful for search tasks like finding tools they were less familiar with, organizational tasks like organizing layers, and other tasks where hands-free input allows them to keep their attention on their composition. Our evaluation allowed us to identify needed technological innovations, such as optimized language models and custom grammars. Other challenges related more directly to use. These include discoverability, acceptance (e.g., encouraging the use of speech in a non-traditional environment), and execution gulfs [45] around valid and invalid natural language (e.g., what end users wanted, or expected, to be valid).

Our contribution includes the proposal of vocal shortcuts for expert-focused creative applications. Through interviews we identified the situations in which speech input as a shortcut technique was *desirable*. Our implementation of VoiceCuts allowed us to understand where speech input could and would be used in more realistic practice.

2 RELATED WORK

Expert shortcuts

Various interaction techniques have been devised to optimize the *expert* experience. While not all have been adopted, several have seen their way into deployed systems. For these optimizations, often the 'physical' costs (e.g., the physical interaction necessary to activate a feature) or 'cognitive' costs (e.g., the learnability, discoverability and development of procedural memory) outweigh the baseline costs of accessing the feature. Solutions such as marking menus (and their variants [32]) can be used to speed access through menus, but with increased complexity may become slow or prone to errors [31]. Spatially organized commands (e.g., CommandMaps [51]) show benefits to expert users, but for complex applications, screen space limits the number of commands that can be displayed.

More conventionally, interfaces offer various keyboard shortcuts (e.g., ctrl-C, ctrl-V, ⌘-⇧-W) or keyboard-based navigation (e.g., alt-F S to open the file menu followed by 's' to save—a sequence familiar to many Windows users). To support a broader set of shortcuts, some keyboards and applications use a large set of modifiers. The LISP Machines 'Space-Cadet Keyboard' famously had seven modifiers [36]. Because the space of possible combinations is enormous, many have focused on teaching end-users key combinations [8, 39]. While some shortcuts are mnemonic in nature ('C' for copy or 'B' for brush), this approach is invariably limited ('C' can't simultaneously be used for cut and copy). IconHK [19] strengthens connections between keys and commands by embedding visual clues in on-screen icons.

Other visual and auditory feedback techniques [20] can improve recall (with some evidence that repeating the command through the auditory channel helps).

Specialized keyboards offer an alternative way to access shortcuts. They range from hardware with dedicated buttons for common shortcuts (e.g., [41]) to keyboards that are aware of which finger is used to press the key [61] and keys which can be pushed in various directions to invoke shortcuts [4].

Adaptive [15, 17, 27] and adaptable interfaces can also reduce cognitive and physical access costs to commonly used features. However, end-users tend not to want to pay the upfront costs for creating optimizations [38]. The alternative, automated adaptation, may challenge experts who may find dynamically changing menus and toolbars to be a hindrance to 'flow' rather than a benefit [15, 34, 58]. Where there is uncertainty in the system's understanding of a shortcut or optimization [23], as in the case of gestures (or in our case speech), end-users may limit their behaviors based on what they think the system can do or how the system will understand their actions [46].

Speech-based shortcuts may reduce both physical and cognitive costs, making their use more attractive. For end-users that cannot easily access a keyboard—for example by using large tablet form factors (e.g., a Wacom screen) or even small forms (e.g., an iPad)—vocal shortcuts may be advantageous. Cognitively, vocal shortcuts may be easier to learn and recall as they are more naturally 'bound' to the feature. Because the uttered command is the same as the feature name, the shortcut is *directly* connected (e.g., 'cut' is 'cut'). Other approaches are *indirect*, relying on mnemonics (e.g., 'ctrl-X') or physical procedural memory (e.g., gestures), and may require acts of mental 'translation.' That said, vocal shortcuts may have increased cognitive costs as the end-user must recall the name and utter it, both potentially unfamiliar steps. Our research thus focuses on identifying where and when vocal shortcuts may be appropriate.

Speech interfaces

Rather than depending on complex keystrokes or novel hardware, speech provides an alternative solution for expert access to tools. Indicators for the benefits of speech input include cases where hands and/or eyes are busy, where access to the keyboard or screen is limited (e.g., mobile applications), where accessibility is a concern, and where speech or natural language is the application's preferred method of interaction [10]. We argue that with proper design, speech is appropriate for expert tasks but that existing solutions may not be directly adaptable.

For example, speech is used for universal accessibility in desktop environments. But because such systems are often generic to all desktop applications, they are rarely optimized for expert use and are often 'retrofitted' on top of existing applications (e.g., [62]). This has the benefit of rapidly providing universal access but may not take expert workflows or application's document model into account.

Historically, there is evidence to suggest that vocal *shortcuts* may benefit experts, as viable speech interfaces have been developed across multiple domains including: spreadsheets [25, 43], word processing [28], programming [5], information visualization and analytics [12, 18, 54, 56], robotic control, and in medical applications [26, 30, 42]. Broadly, these approaches have not focused on *optimizing* the expert experience but rather providing a Natural Language Interface (NLI) to replace more standard interaction modalities.

Speech, and specifically speech-to-text, has certain limitations both for the system (uncertainty and failures in interpretation) and for the end-user.

Research in multimodal interfaces has led to a number of design guidelines for speech to overcome the limitations (e.g., [47, 55]). We leverage this literature in identifying guidelines for our work (e.g., cost models such as those introduced by Baber and Mellor [2]).

At a high level, the tradeoffs between speech-centric and non-speech techniques can be summarized as visibility versus discoverability. Since most non-speech techniques use visible elements (a keyboard, a toolbar, etc.) to accelerate task performance, the user is more likely to accomplish the task, even if she doesn't know about the exact command. However, in the context of creative applications where the user focuses on the canvas object in the center, the visual component can distract the user by creating spatial offsets (e.g., CommandMaps [51]). On the other hand, a speech-centric approach provides no visual cues for commands. Thus the user must 'discover' the command from memory or through trial-and-error. However, we speculate that when users have high levels of expertise, their language models may map directly to the system's interface. Furthermore, the user can likely formulate one or more commands at once without being limited by the visual elements.

Creative applications

Speech-based interfaces for creative applications have been a focus of research for several decades. Early examples include drawing applications (voice-driven MacDraw [48, 49]), 3D object manipulation [6], and GUI design [1]. These systems demonstrate the viability of natural-language as an interaction technique but also highlight key challenges. Research on speech, most often, has identified novice end-users as a target audience (e.g., [44]). However, performance improvements (e.g., time, input) have been shown more broadly. In a tool with restricted vocabulary, Pausch and Leatherby [48] determined that voice improved performance over 'accelerator keys' and that the benefit to experts was greater. Other advantages of speech include enhanced focus on creative tasks.

A study of sketching tasks (e.g., free drawing, illustrating) in a speech augmented drawing system showed qualitative evidence that participants can indulge more in the creative process by issuing necessary commands by speech [53].

While voice solutions have focused on discrete command-and-control operations, a notable exception is VoiceDraw [21], which provides continuous input through sounds (e.g., sounding out vowels to indicate direction). PixelTone [33] supports more general photograph manipulation on mobile devices through a combination of speech and gesture input. Multimodal approaches are warranted in creative applications as deixis is a particular challenge [9, 47, 52]. Deictic phrases (e.g., "put that *here*") are difficult to interpret without additional context.

3 FORMATIVE INTERVIEWS WITH EXPERTS

We interviewed design professionals to learn how they imagine invoking speech commands to enhance their experience. We sought to gather use cases and better understand expert practice and willingness to incorporate speech input into their workflows.

We recruited ten creative experts through a distribution list at a large software company. All interviewees identified as professional designers (7) or artists (3), with experience ranging from 3 to 15 years (M=8.0, SD=4.1). On average, interviewees reported using 4 different creative applications (SD=1.2). Each interview lasted one hour. Six interviews were in person and four were virtual using video and screen sharing software.

Each interview began with background questions to understand the creative expert's professional history and current projects. We then discussed opportunities for speech input in their work in the context of a current project. We asked them to open a recent file and point out situations where they thought they would want to be able to say a command. Then we asked more specific questions about their use of keyboard shortcuts and how they combine stylus, keyboard, and mouse interaction. We also asked what they would say to invoke speech in the scenarios they described.

Suggested use cases

Our participants were excited by the possibilities of including speech interaction in their day-to-day work. They suggested a number of ways that speech input could optimize their work. Broadly, desired features focused on optimizing existing workflows around operations that were viewed as 'costly.' Although interviewees emphasized uncommon 'big' expenses when reflecting on their past behavior (a type of availability bias), we also tried to elicit common, potentially 'small' behaviors that may have large costs in aggregate. Both situations lend themselves to enhancement through shortcuts, speech or otherwise.

Finding infrequently used commands—All of the people we interviewed mentioned wanting to use speech to access commands they do not use frequently. Six out of ten interviewees (E3, E6, E7, E8, E9, E10) relied on keyboard shortcuts for frequently used commands but noted that they only know a handful of keyboard shortcuts and that they vary across software applications. Speech input was perceived as a viable solution for finding less familiar commands quickly. Such a feature has the possibility of reducing cognitive cost of access. Notably, an analogous behavior is by expert users of search interfaces who learn short 'navigational queries' that they know will produce the desired result at the top of the result page [57].

Switching and making brushes—Four participants said they regularly paint digitally (E1, E3, E9, E10) and that they would like to use speech to switch brushes. Searching for the right brush from the typically large collections that experts maintain (e.g., 40-100) is burdensome. Any interaction that requires a keyboard, like searching by keyword or naming a new brush, would be easier to do with speech input because it would allow them to continue holding the stylus rather than switch to a keyboard.

Additionally, several participants said they make their own brushes, and one of them described how he made a business of making and selling brushes (E9). E9 said that when he makes brushes he does multiple explorations and would like to be able to fluidly change parameters as he is drawing strokes. Speech input would allow him to keep his arm on the canvas while changing parameter values, rather than moving back and forth between drawing and changing parameters in a panel.

Working with complex design documents—All of the experts described working with complex documents that include many layers grouped in various ways. As an example, one of the files we saw was a poster showing the stages of product adoption using groups and subgroups of illustrations, text, and charts. The designer (E4) said she worried that she would inadvertently change an element she did not mean to change in moving between the canvas, where she edited the design, and the layers panel, where she locked and unlocked groups. E4 said it would be much easier if she could point to the specific object in the canvas and say "edit layer X" and the system would identify the element of interest (independent of its grouping in the layers panel) and then unlock it and lock the rest.

Using color libraries—Four of the experts suggested using speech input to select colors using semantic color names. One expert (E1) described using different color palettes for a different project and wanting to switch colors using the project color names. For example, he described regularly using color names like "marketing blue", "header blue", "checkout yellow" in projects and communications with the team's

developers. He reported frequently needing to switch between designated colors, for which he refers to a file with the palette, and the associated hex codes. He said he would prefer to be able to integrate these project color names into the interface and switch colors by saying the name.

We note that the latter three use cases, around brushes, design documents, and color, have a similar focus. It is common in creative applications to create 'named' objects (features, tools, layers, etc.) as a way of managing complexity. However, existing shortcut mechanisms such as keyboard shortcuts often ignore user-specific names. While one can easily select the brush tool ('B' in Photoshop) it is not as easy to access the '5 pixel wet ink watercolor brush' (perhaps 'wet 5' as a speech shortcut). Speech input can address this limitation both because it supports a broader vocabulary but also because *what something is called* directly maps to *what is said*. We argue that this aspect of vocal shortcuts has the potential to reduce cognitive costs.

Finally, our participants spoke to the value of speech input for ergonomics and multi-tasking, which are well established as motivations for speech interfaces. E10 said he had bad carpal tunnel for several months and felt that he could use speech input to alleviate the physical stress on his body. E3 suggested using speech input to control secondary tasks like listening to music, web browsing, and email so his hands could stay engaged with the primary creative task.

From expert practice to design goals

Our participants also provided us with a broader sense of their workflow and work environments. Many of the characteristics of the experts that emerged were helpful towards suggesting high-level design goals.

Minimize disruption to creative flow. Related to their desire for efficiency, our participants all described that a viable speech interface would need to support the expert's flow. E2 and E4 said they found it problematic to change posture and eye focus in the midst of some creative tasks like drawing or editing photos.

Support flexible device/application configurations. Our participants described how they *tend to use a variety of input devices*, including a desktop and mobile tablets. Seven of our interviewees (E1, E2, E4, E5, E8, E9, E10) use tablets and stylus pens. The stylus pen supports drawing tasks but limits the use of keyboards, as one hand is always holding the pen. Additionally, our participants said they *use multiple applications in their professional work*, which requires them to remember *different* shortcuts for *similar* tools.

Provide support for user customization. Indicative of their potential to adopt speech, our participants expressed having *strong incentives to identify ways to make their workflow more efficient*. Six interviewees (E3, E6, E7, E8, E9, E10) heavily use keyboard shortcuts to expedite their work process. While prior research characterizes experts as resistant to new technology, since in many cases they have already optimized their workflow [3], all of our interviewees indicated that if speech input provides enough efficiency and helps them focus more on the creative task, they are willing to learn and change their work practice to be more productive and creative. Specifically, customization supporting referencing names for tools, colors, and brushes was of high interest.

4 SYSTEM DESCRIPTION

We demonstrate the viability and limitations of vocal shortcuts with a prototype implementation, VoiceCuts. VoiceCuts was built to add a speech modality to the creative application, Adobe Photoshop. The system contains four main components: a speech input interface, a command interpreter, an execution engine, and a panel for customizing commands (Fig. 2). As the user speaks, the system turns the audio into text, translates the command to one or more application operations, and executes them. Our prototype currently uses the Google Speech API [50] to convert audio to text and leverages the Adobe Photoshop scriptable execution engine to perform the identified operations. All interface components are implemented using JavaScript as an extension to Photoshop. The 'engine' of VoiceCuts is the interpreter that leverages application, document, and user context.

Because our goal is to minimize disruption to creative flow, VoiceCuts is designed with a command-and-control approach: the user utters a command and the application executes it. VoiceCuts does not talk back and does not support conversations. It was designed and built to expect short phrases and partial commands. To encourage speech interaction and support creative flow, VoiceCuts always tries to 'do something' even in the face of uncertainty.

VoiceCuts user experience

Let's follow Stefanie, who is a professional web designer, as she works on a painting with VoiceCuts. Stefanie loves her large Wacom touch display (see Figure 3) even though she can't easily use her keyboard. She clicks the button on her stylus pen and speaks the vocal shortcut "watercolor brush." VoiceCuts opens the listening panel where it shows Stefanie's words (Fig. 2(a)). A blinking microphone icon tells Stephanie that the system is listening. VoiceCuts waits for Stefanie to stop talking, interprets her command, and selects the watercolor brush. To change the size of her brush, Stefanie can use the vocal shortcut: "size 50" or simply "50." VoiceCuts remembers the most recently changed parameters for each tool and makes it easy to change them without having to repeat the parameter name.

Stefanie also likes to use VoiceCuts's support for customization. She has some favorite colors, *Fall grey* and *sunrise*

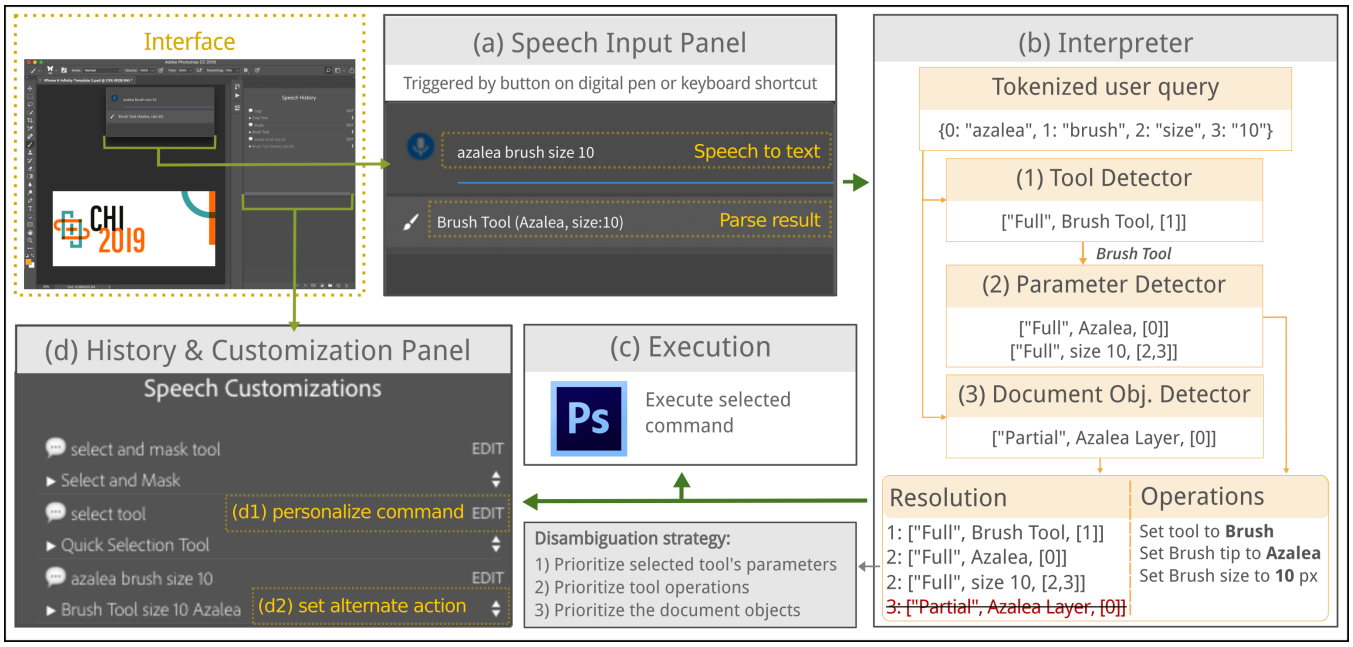

Figure 2: VoiceCuts's interface and architecture, illustrating how speech interaction is supported throughout the system.

blue, that she set in VoiceCuts. Adding new customizations is easy. Stefanie can use the interactive customizations and history panel (Fig. 2(d)) or add the commands in a text file.

To select parts of her painting, Stefanie clicks her pen button and says "select tool" (intending the *select and mask tool*). VoiceCuts misinterprets what she means and gives her the *quick selection tool*. To get the right tool, Stefanie uses the full name: "select and mask tool." To make things easier for next time, she sets the vocal shortcut keyword for "Select & Mask" to "select" using the Edit button (Figure 2(d1)). When a shortcut ambiguously maps to multiple matches, VoiceCuts takes a 'greedy' best-guess approach. It picks the most likely command, but lets the user switch to a different interpretation using a drop-down menu (Figure 2 (d2)). Corrections influence subsequent interpretations.

Interpreting user commands

The VoiceCuts interpreter (Fig. 2 (b)) uses the output from the Google Speech API to map the vocal command to corresponding application operation(s). The prototype interpreter supports three types of operations: activating *tools or features*, setting *tool parameters*, and selecting *document objects* (e.g., image layers or groups).

The interpreter tokenizes the vocal shortcut, removes stop words, and passes the text through *tool, parameter, and document object detectors*. If more than one detector finds a match or one detector finds multiple matches, the interpreter resolves the ambiguity using a set of heuristics and selects one or more final operations. We prioritize the user's current context and prioritize parameter matches for the currently selected tool over operation and document property matches. Even when detection confidence is low, VoiceCuts executes the operation and logs the user utterance and corresponding operation in the customization and history panel.

All detectors search for full and partial matches by comparing each token in the vocal shortcut to dictionaries that contain the names of the operations, parameters, and document objects. For example, if the user command is "quick selection tool," then the tool detector will return a full match to the "Quick Selection Tool." To find partial matches, the detectors compare all possible n-grams (sequential subsets of n-words) to the dictionary items. For example, "selection tool" is a partial match with "quick selection tool" and "path selection tool," while "select tool" is a partial match with "Select & Mask." The detectors return the matches that yield the longest matching n-gram. If multiple matches are possible, all are returned. For example, for the command "select tool," the tool detector returns "path selection tool", "quick selection tool", and "select & mask." As is evident from this example, we support variants by matching on word stems (e.g., *selection* becomes *select*). A more restrictive grammar–one that does not support 'fuzzy' matching at all–may yield more precise results. However, we used the fuzzy matching approach to encourage the experts to try speech without worrying about learning the grammar.

Tool and feature detector: The tool detector compares the shortcut command to all menu, toolbar, and macro operations. A pre-defined set of command names was developed in a pre-processing step by scraping tool and parameter names

from the application. Additionally, VoiceCuts dynamically queries the application on startup for any custom tools, such as macros or tool presets, created by the user or downloaded from the web.

Parameter detector: The parameter detector compares the user command to the parameters of the currently selected tool. If the user utterance specifies a tool/feature and a parameter change simultaneously (e.g. "butterfly brush size 10"), the parameter detector compares the user command to the parameters of the tool specified in the utterance. To generate a list of tool parameters, VoiceCuts dynamically queries the application and builds a list of *numeric parameters* (e.g., size, opacity) and *text parameters* (e.g., brush name, color name).

Numeric parameters include units (pixels, %, etc) and a possible value range (e.g., size: 0-1000). Text parameters include a simple list (e.g, brush name: *butterfly*, *azalea*, *watercolor*, etc).

Identifying text parameters: When a tool has a text parameter, the detector compares the user command to a list of known parameters for that tool. For example, when the reference tool is the *brush*, the user can specify a *brush tip* parameter (e.g., "watercolor brush", "oil pastel large brush").

Identifying numeric parameters: The majority of tools have numeric parameters. To support numeric parameters, the parameter detector first finds all numeric values in the command and then tries to associate them with the right parameters. It is flexible to several formulations including:

- parameter name, numeric value, units (e.g., *size 10 pixels*)
- parameter name, numeric value, no units (e.g., *size 10*)
- numeric value, parameter name (e.g., *10 size*)
- numeric value, units, parameter name (e.g., *10 pixels size*)
- numeric value (e.g., *10*)
- numeric value, units (e.g., *10 pixels*)

When VoiceCuts finds a numeric value, the parameter detector searches for a unit and parameter name in ± 2 token windows around the numeric value. This allows the user to disambiguate parameters to the system when the tool has multiple numerical parameters (e.g., '10,' with no units or additional information, could refer to opacity or brush size).

More specifically, VoiceCuts disambiguates using one of three methods: from units in the command, from a previous command that changes a parameter, and from frequency of use. If the unit is specified, the detector chooses the most frequently used parameter with the specified unit (e.g., "size" for "pixels"). If neither the name not the unit are specified, VoiceCuts infers the parameter from the most recent command. For example, if the user command is "20" and the most recently edited parameter is opacity, VoiceCuts will select the opacity parameter. If there is no history of changing a parameter, VoiceCuts chooses the most frequently used parameter for the tool (e.g., size for the brush tool). To process a command containing multiple parameters (e.g., size 10 opacity 20), the detector repeats this process for each numeric value in the command.

Document object detector: The document object detector compares the user command to the current list of layers and groups of layers. The list of layers is built dynamically given the file or editing activities.

Disambiguation: As each detector works independently, ambiguous results are possible. For example, if the user is painting with a brush and says "crop," the tool detector returns "Crop tool" and "Image>Crop", the parameter detector returns a brush named "kid crop 4", and the document object detector returns layer "crop". These results are all using the same word in the command for their interpretation, "crop." If there is no overlap, all operations returned by the detectors are executed in the following order: layer selection, tool change, parameter change. So the command "watercolor brush on the sky" will select the sky layer, select the brush tool, and set the brush tip to watercolor. When there is overlap, VoiceCuts uses heuristics to select among the operations: Full matches to tools, parameters, and document objects are preferred over partial matches. Special words like tool, layer, and panel give weight to the corresponding detector. Finally, VoiceCuts disambiguates using the specificity of the context with document context being most preferred, followed by user context, parameter context and finally application context. So in the "crop" example above, VoiceCuts selects the crop layer.

Customizing vocal shortcuts

VoiceCuts supports adding custom vocal shortcuts so that experts can use words that are most meaningful and best for them. VoiceCuts checks custom shortcuts first before running the interpreter pipeline. Custom shortcuts can be specified with a text file or interactively through the History and Customization Panel (see Figure 2(d)).

Limitations

Making use of a general purpose speech-to-text engine, such as Google's speech-to-text engine, allowed us to focus on the interpretation engine. Unfortunately, general speech-to-text engines are not designed for short commands. Because of the way they are trained, they often need longer sentences to perform well or might expect users to engage in conversational or more formal language. Thus performance for our specific application will not be as good as general reported performance for speech-to-text. Short commands have a lot more ambiguity than longer sentences, and accuracy on each word is more critical. In the context of expert users who want to stay in the flow of their task, high transcription accuracy

Figure 3: The device configurations for the lab study.

is especially important. We expect that a custom trained speech-to-text model will help improve performance.

VoiceCuts is also limited by the Photoshop execution API. There are some commands that are inaccessible. For example, we can't support comparative commands like "bigger" and "smaller" even though shortcuts for these commands exist. We also can't control modal dialogs through speech input. These limitations can violate user expectations, because Photoshop does have shortcuts for comparative commands and navigating dialogs. Potential fixes may include gaining access to the program's source code, using accessibility APIs or external drivers (e.g., [59]).

In the current VoiceCuts implementation, all tools and menus are treated equally, but in reality there are new efficient workflows and older outdated workflows. For example, image adjustments are better done through non-destructive adjustment layers. The adjustment names are exactly the same, leading to ambiguity. Which "Hue/Saturation" does the user want? Destructive or non-destructive? The VoiceCuts interpreter could be improved by weighing some tools higher than others. This could have the additional benefit of showing users new features they may not be aware of (in the style of [40]).

5 EVALUATION: LAB STUDY

To gather feedback from a larger group of professional creatives, we conducted an exploratory laboratory evaluation. Our goal was to assess the potential of vocal shortcuts and test the design and implementation of VoiceCuts.

Participants

Eight participants (four female; seven native speakers) were recruited through an HCI-focused email list at a large university. We required that participants had 5 or more years of Photoshop experience, used it at least 1 to 3 times a week, had experience with a stylus pen, and identified as an intermediate to expert user of Adobe Photoshop. The participants received a $40 Amazon gift card.

Study Procedure & Task

We conducted all sessions in a conference room and the sessions lasted from 1 to 1.5 hours. We used a Wacom Cintiq 27QHD Touch Display and Creative Pen (Fig. 3 (a), (b)) and recorded the screen as well as session audio. We provided a keyboard (Fig. 3(c)), a Wacom ExpressKey (Fig. 3(d)) or a button on the stylus pen for participants to use to trigger speech interaction. We located the ExpressKey on the non-dominant side in case a participant wanted to trigger interaction while using the pen with their dominant hand.

Because the laptop running VoiceCuts was moved aside to make space for the Wacom, we placed an external microphone to listen to the participants' voice. This ensured high quality logging of the session audio but likely improved speech-to-text quality as well (Fig. 3(e)). To provide a more ecologically valid work setting, we instructed participants to provide their own tool settings, which we installed ahead of time.

At the start of each session, we showed participants how to use VoiceCuts through some examples of vocal shortcuts. After testing these commands, we asked participants to work on a design task for 45 minutes (spending roughly 15 minutes planning a design and 30 minutes creating it). Specifically, we asked participants to create an invitation for a harvest party (Fig. 4). To give them a chance to use a range of tools, we required participants to edit a given image and to make one or more illustrations. We encouraged participants to use speech input whenever they thought it might help even if they were not sure whether the command would work.

Figure 4: A sample result of the task from P3.

After completing the task, we asked participants to share their impressions of the overall speech interaction experience. We posed specific questions including, "How do you think that speech input affected the flow of your work?", "What was your biggest frustration?," and "What did you think was the biggest win for using speech input in your session?" We also asked about their ideal speech interface. To understand why participants used specific vocal shortcuts and the expected Photoshop operations,

we asked follow-up questions about their intent. Finally, we discussed their willingness to customize commands.

Findings

Overall, all participants were able to complete the task and expressed a willingness to adopt vocal shortcuts into their workflow. On average each participant uttered 60.5 (SD=18.6) speech commands over 30 minutes. Each command consisted of 1.4 words on average (SD=0.93). Together the participants uttered 484 commands. The participants appreciated the capability to issue partial queries, such as setting a parameter within a tool space (e.g., "size 10") without specifying the name of the tool. Participants issued 105 such partial commands, or 22% of the total vocal shortcut use.

VoiceCuts performed as expected for 82% of commands. However, 62 out of 484 utterances (12.8%) had speech-to-text errors and 25 (5.2% of total speech commands) were misinterpreted or resulted in errors during execution. Ten of the 25 were comparative commands like "bigger," "larger," and "5 more" The remaining 15 commands were to open dialogs. P2 said *" I would just say to "change opacity" rather than clicking a tiny 30 px button up here"*, referring to how she wanted to open the panel with opacity controls prior to actually setting the new opacity value. Participants wanted to open the dialog because they weren't always sure of the correct parameter value.

Speech as fast (navigational) search: We observed participants using speech as a faster way to search for a tool they wanted when they couldn't locate the tool with the GUI or didn't know the correct keyboard shortcut. P1 was looking for a tool that he could vaguely remember the name of and couldn't locate by searching the toolbar. He instead opened VoiceCuts and said the name of the tool, but first noted to the experiment administrator: *"There was a tool called color bucket. I was not sure where that was, so I quickly tried."*

P6 similarly invoked "swatches" by speech, describing how he wanted to say it because he didn't know the shortcut. P3 used speech input to search for an operation for which she did not know the exact name: *"I just checked whether the same function exists in Photoshop as Adobe Illustrator. It felt faster to search by speech than navigating the interface".*

Speech to organize element hierarchies: The complexity of their work and motivation to be efficient makes expert users of creative applications more likely to structure their creative process [29]. This can result in, for example, organizing their workspace. Despite the short time limit, many participants spent time during their design task to organize layers, and leveraged speech to aid in this work. Of the 484 speech commands, 68 (14.0%) were to organize and manage layers and group hierarchies. Some participants (P2, P3) mentioned that using speech for organizing commands seemed to help them understand the structure better. P3 said *"by verbalizing the speech command 'the default text layer and leaves layer in group 1', I can make sure that's the structure that I want".*

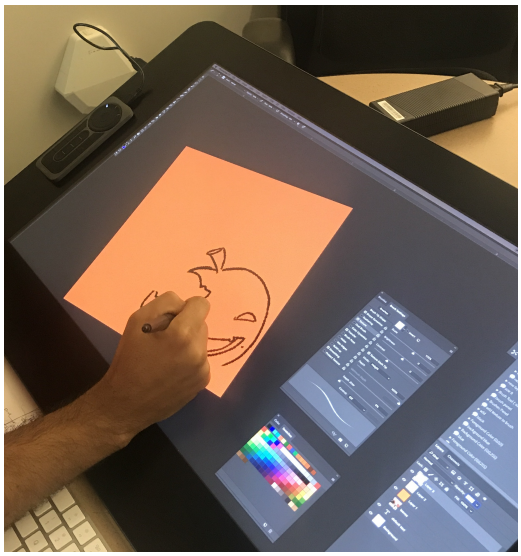

Figure 5: P8 tilted the canvas to replicate the experience of drawing in a sketchbook.

Multiple participants (P2, P3, P5, P6, P7, P8) also envisioned even bigger benefits from speech input in cases where the layering and grouping structures they created might be more complex. P5 mentioned, "Imagine selecting a layer among hundreds that are stored in ten different groups." She found ordering groups and layers by speech input especially useful because she always has a clear idea about the organization she wants: "it is easier to say 'put group 1 on top of group 2' than dragging the groups around because that is how I envision two groups should be layered."

Speech as a 'posture-free' input modality: Three of our participants (P2, P3, P7) had more than 10 years of work experience as professional designers. They mentioned that speech input could be useful during occasional bouts of occupational illness that they and other designers experience. P3 mentioned, *"In our team, we regularly exchange tips for preventing carpal tunnel syndrome and shoulder pain. Speech interaction can release those pains".* A few participants (P4, P8) mentioned the importance of posture to their process. P4 mentioned, "I like to lean on the tablet and draw. It would be nice if you don't have to change postures to access the keyboard and focus on work." P8 also mentioned that speech input will be handy especially for illustrators who believe that certain postures are more conducive to their creative flow. *"When I draw, I tilt my whole canvas in Photoshop so that I can maintain a natural posture like drawing on the sketchbook. I think speech is really handy, if I am in this mode."*

Speech as a macro: We observed that participants often associated speech input with accomplishing multiple actions

automatically. For example, P1 wanted to execute three consecutive operations that he often does in a row and hoped he could execute them all at once with VoiceCuts. He mentioned, *"it may be just my expectation for speech interaction to automate my task."* P2 mentioned, *"I hope I can set up the document with exact resolution and other settings at once instead of pressing keys like command, option and I, and again going through multiple drop-downs."*

Speech for facilitating design exploration: We initially expected that speech input might help users when they have a specific intention in mind and be less useful during more exploratory work when the user is uncertain of the best parameter setting. However, three participants (P5, P6, P7) said speech could be helpful for exploring different visual styles. For example, P5 described how he would typically have to invoke a menu, then noted that *"instead of having to go through those hoops, I can just say 'brighter,' and my eyes never leave my work. So I can just focus on that and see how it looks instead of ... where is this menu ... and fiddling with it."*. P6 mentioned, *"I have to play with a bunch of different numbers over and over again. The drop-down menu, and having to sift through all of that. So to me, being able to sit there and look at it and say 'larger,' and 'smaller,' that would be so much more helpful."*. In selecting a font, a user has to go back and forth from the menu bar and the text to assess the visual style of the text. P7 mentioned that speech input could be useful for selecting fonts: *"It will be super handy if I can just say next next next when I select the font."*

Bias from previous experience: Although the participants were told that VoiceCuts was aware of the application context and expected short commands, we did observe participants adding specificity to some of their speech commands under the belief that it would help the system understand it. For example, P1, P5, P8 always said "name of the tool" + "tool" to activate a tool, instead of saying only the tool name (e.g., saying "brush tool" instead of simply "brush"). Many participants (P1, P8) also added "layer" after a layer name to select a layer, (e.g., "select leaves layer" instead of "select leaves"). Since most participants (except P8) said they had experience with voice assistants (e.g., Siri, Alexa), they felt they should specify the command to be understood by our prototype as they did for those assistants. P1 mentioned that once he had gained more confidence using the system and observing its ability to process short commands, he would likely change the way he issued speech commands to take advantage of shorter phrases.

Customization: None of our participants customized the speech commands during their session. However, in our follow-up interview, all participants said they would customize speech commands assuming longer-term use. Customization examples they provided all focused on command shortening (e.g., 'paint bucket tool' to 'paint').

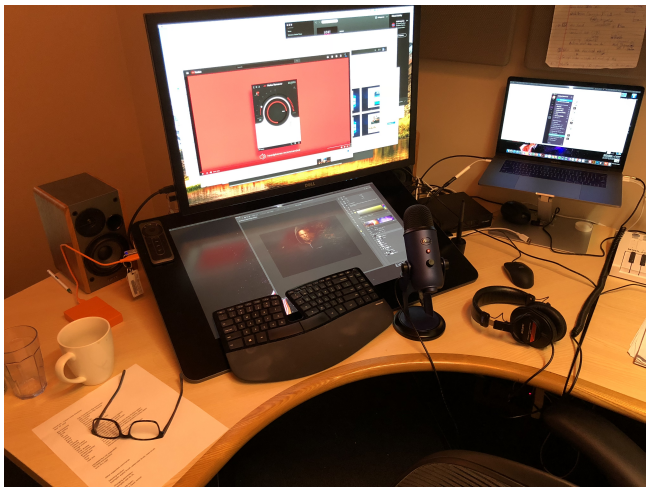

Figure 6: Our case study professional designer uses a large monitor, a Wacom touch display with a wireless keyboard, stylus pen, a mouse, a laptop, and a microphone as part of his work.

Given this feedback, a natural next step was to deploy VoiceCuts and see its use in a real-world scenario where people can use it in the context of their own environment.

6 EVALUATION: CASE STUDY

We deployed VoiceCuts with one professional designer at a large software company and collected his feedback from three real-world usage sessions spanning 4 hours. Figure 6 shows his desktop setup with a large monitor, a Wacom touch display, a laptop, a wireless keyboard, mouse, stylus, and microphone. In an initial session, he tried out the prototype with some success but felt that it would be much more useful to him with custom vocal shortcuts. In a second session, he walked the authors through the application describing 23 commands he would like to be able to say that would be helpful to his work. For example, he wanted to use 'g-blur' for 'gaussian blur' and 'select' for the 'Rectangular Marquee Tool.' Of the 23 commands he requested, 10 were already supported by VoiceCuts, 11 required custom specification, and 2 could not be supported due to technical limitations of the Photoshop extensibility API. Additionally, we specified homophone shortcuts for vocal shortcuts that were not well supported by the speech-to-text engine. For example, "hue" was most frequently transcribed as "hugh" and "cute," and "fill" was most frequently transcribed as "phil." An unintended side benefit of supporting customization is that it can help with correcting common speech-to-text errors. In a final logged session, the designer used the customized prototype and provided additional feedback through email.

Overall, the designer's feedback was positive. He found value in vocal shortcuts and felt that VoiceCuts saved him time. From system logs, we can see that he issued 58 different commands and the system took action on 33 of them. Successful commands included adding layer styles like a drop shadow and color overlay and transforming and rotating objects. Commands that did not work were due to technical limitations of the current prototype, like supporting text entry through voice, and speech-to-text transcription mistakes ('rasterize' became 'restaurant', and 'wrestler eyes,' was 'Rochester eyes').

The designer also had some suggestions for improvement, most significantly around speed. He found that the pause time while the system detects silence was too much. Also, he wanted to train VoiceCuts himself so that it could accommodate his accent and make fewer speech-to-text errors. Finally, he wanted to be able to do text entry through voice, so he would not have to switch to his keyboard.

7 DISCUSSION AND FUTURE WORK

The feedback from our nine expert participants (eight from the lab study and one from the case study) confirmed that vocal shortcuts are useful additions to their practice. All were excited by the possibilities of improving their workflows by leveraging speech input in concert with other devices. However, our study also surfaced concerns about the interplay between the underlying technology and the UX. These prevented the system from uniformly lowering the physical and cognitive costs of vocal shortcuts in a way that made them preferable to existing interactions.

When to listen

In contrast to our expectation, all participants but one (P4) wanted a speech system that was *always* listening. Participants described how they tended to have a quiet and private working environment and found triggering speech input by pressing a button distracting. P7 said *"I definitely don't think a button should be used. If you're trying to eliminate using buttons and using your hands as much as possible, that's definitely a step in the right direction."* Always-listening interfaces are possible and will likely reduce activation cost. However, further research will be necessary to build a mechanism for reliably separating commands to an application from other speech (to a colleague, on the phone, from a video playing at the same time). This may include investigating the types of audio present in modern work environments. Using an activation 'hot-word' to indicate the start of a command (as with Siri or Alexa) is a possibility, but does 'lengthen' the shortcut. Identifying the right balance between fast invocation and accurate 'listening' is an important open question.

Another aspect of listening performance is the speed of execution. The current VoiceCuts prototype uses a remote speech-to-text engine, which takes 1 to 2 seconds. The majority of this time is due to (1) connecting to the remote server to convert the speech to text and (2) detecting silence to know when the user is done talking. This performance can be improved by using a local engine or a hybrid local-server approach [33]. To reduce the time that the parser waits to detect silence, future work can explore a multimodal approach to triggering execution where the user touches the canvas to indicate he/she is done talking.

Customization support

It is clear that customization is a key part of making vocal shortcuts work well. But how to support customization in a way that is lightweight and part of existing workflows is an open question. Programming by demonstration approaches [37] or mixed-initiative personalization [7] may offer potential directions and help address the challenges of open-vocabulary speech-to-text transcription. While VoiceCuts's speech-to-text performance could be improved with custom vocabularies and custom speech-to-command models, it's worth considering whether a good customization interface can help address some of these errors. Perhaps through a few spoken examples, users can define their own speech-to-command mappings. A hybrid approach may be the best path forward, as text entry and parameter setting would still require a speech-to-text transcription engine.

Costs and mistakes

Vocal shortcuts can benefit expert users in many ways, but also introduce a new set of costs: error costs. While a user may type the wrong hotkey, traditional shortcut techniques are both deterministic and largely error-free. Speech-to-text and command parsing both introduce errors that can slow use, and by extension, adoption. Different errors also might have different costs. Switching to the wrong tool may be easy to correct, but executing an expensive filter (which requires an extra undo) is more costly. We argue that vocal shortcuts could be improved by maintaining a more formal cost model not unlike mixed-initiative cost/utility models in [24]. Improving the design of the UX experience in light of failures is also critical.

Generalizability

We tested our approach in one software application, but point to similarities between Photoshop and many other creative applications that suggest it may generalize more broadly. Beyond Photoshop many creative applications use a tool, parameter and document object paradigm (Bohemian Sketch, Microsoft PowerPoint, etc). Our approach supports these three operations with distinct detectors. Applications that follow a model of 1) selecting target objects to manipulate (e.g., shapes, text elements in PowerPoint, cells in Excel), 2)

providing tools to change the properties of the objects (e.g., applying tools the shapes in PowerPoint, styling the text in Word) and 3) allowing organization of object hierarchy (e.g., slides in PowerPoint, sheets in Excel, pages in Word) can adapt our approach. For an application that has operation categories other than these three, our framework can be extended with additional detectors. This multi-detector approach has the benefit that a developer can integrate a new detector that can work "in parallel" and offer both high recall and precision.

Creative flow

Research on creativity underscores the potential for small disruptions to have noticeable impacts on the perceived effectiveness and workflow of a creative expert. Evidence from past work highlights that creative experts rely on a state of flow, or total absorption in their work to accomplish creative tasks [13, 14]. Future work might study how adding speech support to creative applications specifically supports experts' ability to maintain flow. Multiple participants described a desire to maintain the focus of their creative flow and not directing attention away from their composition to the application interface. Interestingly, this also applied to background tasks outside of the creative application. One of our interviewees said that speech could help him stay focused by allowing him to manipulate secondary applications, such as a music player. This participant typically reserves his main monitor for his creative work and his laptop screen for everything else: email, music, web browsing, etc. To change to a different radio station or look up a tutorial to help him with a less familiar task, he has to move his cursor to his other screen. Future work might explore how speech interaction can control secondary applications that have high physical and cognitive cost specifically during creative focus.

Creative collaboration

One of the formative study participants alluded to the potential for speech interaction to support expert creative work in collaborative settings. He mentioned that he wants to interact with the speech-enabled creative applications when he introduces his design work to clients. During such meetings, he often fumbles with the keyboard or mouse when he has to make on the fly design changes. He felt that speech input may alleviate some of the in-the-moment stress and support his collaborative creative process.

8 CONCLUSION

In this work, we describe a novel approach to accelerating expert use of a creative application through vocal shortcuts. Vocal shortcuts are intended to lower the cognitive and physical cost of accessing features in complex creative software. Physically, a keyboard-free use allows the creative professional to focus on their primary tablet interface. A larger, and more mnemonic, vocabulary can make it cognitively easier to learn and recall the shortcuts for a large and complex set of tools and features. We experiment with vocal shortcuts by implementing VoiceCuts as a plug-in to the Adobe Photoshop application. By dynamically mining the creative's current work environment and supporting customization, VoiceCuts supports vocal shortcuts that reference custom names for tools and layers. Feedback from creative experts confirms the potential of this approach and points to future directions for improvement. Our work sheds light on the viability and limitations of speech interfaces today and provides a foundation for the next generation of speech-enabled creative applications.

9 ACKNOWLEDGEMENTS

We thank Celso Gomes and our study participants for their useful feedback.

REFERENCES

[1] Mohammad M. Alsuraihi and Dimitris I. Rigas. 2007. How Effective is It to Design by Voice?. In *Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI...But Not As We Know It - Volume 2 (BCS-HCI '07)*. BCS Learning & Development Ltd., Swindon, UK, 159–162. http://dl.acm.org/citation.cfm?id=1531407.1531449

[2] Chris Babe and Brian Mellor. 2001. Using critical path analysis to model multimodal human-computer interaction. *International Journal of Human-Computer Studies* 54, 4 (2001), 613 – 636. DOI: http://dx.doi.org/10.1006/ijhc.2000.0452

[3] Julie L Baher and Bill Westerman. 2009. The usability of creativity: experts v. novices. In *Proceedings of the seventh ACM conference on Creativity and cognition*. ACM, 351–352.

[4] Gilles Bailly, Thomas Pietrzak, Jonathan Deber, and Daniel J. Wigdor. 2013. MéTamorphe: Augmenting Hotkey Usage with Actuated Keys. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 563–572. DOI: http://dx.doi.org/10.1145/2470654.2470734

[5] A. Begel and S. L. Graham. 2006. An Assessment of a Speech-Based Programming Environment. In *Visual Languages and Human-Centric Computing (VL/HCC'06)*. 116–120. DOI: http://dx.doi.org/10.1109/VLHCC.2006.9

[6] Richard A Bolt. 1980. *"Put-that-there": Voice and gesture at the graphics interface*. Vol. 14. ACM.

[7] Andrea Bunt, Cristina Conati, and Joanna McGrenere. 2007. Supporting Interface Customization Using a Mixed-initiative Approach. In *Proceedings of the 12th International Conference on Intelligent User Interfaces (IUI '07)*. ACM, New York, NY, USA, 92–101. DOI: http://dx.doi.org/10.1145/1216295.1216317

[8] Andy Cockburn, Carl Gutwin, Joey Scarr, and Sylvain Malacria. 2014. Supporting Novice to Expert Transitions in User Interfaces. *ACM Comput. Surv.* 47, 2, Article 31 (Nov. 2014), 36 pages. DOI: http://dx.doi.org/10.1145/2659796

[9] Philip R Cohen, Mary Dalrymple, Douglas B Moran, FC Pereira, and Joseph W Sullivan. 1989. Synergistic use of direct manipulation and natural language. In *ACM SIGCHI Bulletin*, Vol. 20. ACM, 227–233.

[10] P R Cohen and S L Oviatt. 1995. The role of voice input for human-machine communication. *Proceedings of the National Academy of*

Sciences 92, 22 (1995), 9921–9927. http://www.pnas.org/content/92/22/9921.abstract

[11] Eric Corbett and Astrid Weber. 2016. What can I say?: addressing user experience challenges of a mobile voice user interface for accessibility. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*. ACM, 72–82.

[12] Kenneth Cox, Rebecca E. Grinter, Stacie L. Hibino, Lalita Jategaonkar Jagadeesan, and David Mantilla. 2001. A Multi-Modal Natural Language Interface to an Information Visualization Environment. *International Journal of Speech Technology* 4, 3 (01 Jul 2001), 297–314. DOI: http://dx.doi.org/10.1023/A:1011368926479

[13] Mihaly Csikszentmihalyi. 1991. *Flow: The Psychology of Optimal Experience*. Harper Perennial, New York, NY.

[14] Mihaly Csikszentmihalyi. 1996. *Flow and the psychology of discovery and invention*. New York: Harper Collins.

[15] Leah Findlater and Krzysztof Z Gajos. 2009. Design space and evaluation challenges of adaptive graphical user interfaces. *AI Magazine* 30, 4 (2009), 68.

[16] C Ailie Fraser, Mira Dontcheva, Holger Winnemöller, Sheryl Ehrlich, and Scott Klemmer. 2016. DiscoverySpace: Suggesting Actions in Complex Software. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems*. ACM, 1221–1232.

[17] Krzysztof Z. Gajos, Katherine Everitt, Desney S. Tan, Mary Czerwinski, and Daniel S. Weld. 2008. Predictability and Accuracy in Adaptive User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 1271–1274. DOI: http://dx.doi.org/10.1145/1357054.1357252

[18] Tong Gao, Mira Dontcheva, Eytan Adar, Zhicheng Liu, and Karrie G. Karahalios. 2015. DataTone: Managing Ambiguity in Natural Language Interfaces for Data Visualization. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology (UIST '15)*. ACM, New York, NY, USA, 489–500. DOI: http://dx.doi.org/10.1145/2807442.2807478

[19] Emmanouil Giannisakis, Gilles Bailly, Sylvain Malacria, and Fanny Chevalier. 2017. IconHK: Using Toolbar Button Icons to Communicate Keyboard Shortcuts. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 4715–4726. DOI: http://dx.doi.org/10.1145/3025453.3025595

[20] Tovi Grossman, Pierre Dragicevic, and Ravin Balakrishnan. 2007. Strategies for Accelerating On-line Learning of Hotkeys. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07)*. ACM, New York, NY, USA, 1591–1600. DOI: http://dx.doi.org/10.1145/1240624.1240865

[21] Susumu Harada, Jacob O. Wobbrock, and James A. Landay. 2007. Voicedraw: A Hands-free Voice-driven Drawing Application for People with Motor Impairments. In *Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility (Assets '07)*. ACM, New York, NY, USA, 27–34. DOI: http://dx.doi.org/10.1145/1296843.1296850

[22] M Helander, T Landauer, and P Prabhu. 1997. Mental models and user models. In *Handbook of human-computer interaction*. Elsevier, 49–63.

[23] K. Höök. 2000. Steps to take before intelligent user interfaces become real. *Interacting with Computers* 12, 4 (2000), 409 – 426. DOI: http://dx.doi.org/10.1016/S0953-5438(99)00006-5

[24] Eric Horvitz. 1999. Principles of Mixed-initiative User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. ACM, New York, NY, USA, 159–166. DOI: http://dx.doi.org/10.1145/302979.303030

[25] Jim Hugunin and Victor W Zue. 1997. On the design of effective speech-based interfaces for desktop applications. In *Fifth European Conference on Speech Communication and Technology*.

[26] M. G. Jacob, Y. T. Li, and J. P. Wachs. 2012. Gestonurse: A multimodal robotic scrub nurse. In *2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 153–154. DOI: http://dx.doi.org/10.1145/2157689.2157731

[27] Anthony Jameson and Krzysztof Z Gajos. 2012. Systems that adapt to their users. In *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications, 3ed*. CRC Press, Boca Raton, FL.

[28] Lewis R. Karl, Michael Pettey, and Ben Shneiderman. 1993. Speech versus mouse commands for word processing: an empirical evaluation. *International Journal of Man-Machine Studies* 39, 4 (1993), 667 – 687. DOI: http://dx.doi.org/10.1006/imms.1993.1078

[29] Manolya Kavakli and John S Gero. 2002. The structure of concurrent cognitive actions: a case study on novice and expert designers. *Design studies* 23, 1 (2002), 25–40.

[30] Anna Kochan. 2005. Scalpel please, robot: Penelope's debut in the operating room. *Industrial Robot: An International Journal* 32, 6 (2005), 449–451. DOI: http://dx.doi.org/10.1108/01439910510629136

[31] Gordon Kurtenbach and William Buxton. 1993. The Limits of Expert Performance Using Hierarchic Marking Menus. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems (CHI '93)*. ACM, New York, NY, USA, 482–487. DOI: http://dx.doi.org/10.1145/169059.169426

[32] Gordon Kurtenbach, George W. Fitzmaurice, Russell N. Owen, and Thomas Baudel. 1999. The Hotbox: Efficient Access to a Large Number of Menu-items. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. ACM, New York, NY, USA, 231–237. DOI: http://dx.doi.org/10.1145/302979.303047

[33] Gierad P Laput, Mira Dontcheva, Gregg Wilensky, Walter Chang, Aseem Agarwala, Jason Linder, and Eytan Adar. 2013. Pixeltone: A multimodal interface for image editing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2185–2194.

[34] Talia Lavie and Joachim Meyer. 2010. Benefits and costs of adaptive user interfaces. *International Journal of Human-Computer Studies* 68, 8 (2010), 508 – 524. DOI: http://dx.doi.org/10.1016/j.ijhcs.2010.01.004 Measuring the Impact of Personalization and Recommendation on User Behaviour.

[35] Effie L-C Law and Paul Van Schaik. 2010. Modelling user experience–An agenda for research and practice. *Interacting with computers* 22, 5 (2010), 313–322.

[36] Xah Lee. 2017. History of Emacs and vi Keys. http://xahlee.info/kbd/keyboard_hardware_and_key_choices.html. (2017).

[37] Toby Jia-Jun Li, Amos Azaria, and Brad A. Myers. 2017. SUGILITE: Creating Multimodal Smartphone Automation by Demonstration. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 6038–6049. DOI: http://dx.doi.org/10.1145/3025453.3025483

[38] Wendy E. Mackay. 1991. Triggers and Barriers to Customizing Software. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '91)*. ACM, New York, NY, USA, 153–160. DOI: http://dx.doi.org/10.1145/108844.108867

[39] Sylvain Malacria, Joey Scarr, Andy Cockburn, Carl Gutwin, and Tovi Grossman. 2013. Skillometers: Reflective Widgets That Motivate and Help Users to Improve Performance. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 321–330. DOI: http://dx.doi.org/10.1145/2501988.2501996

[40] Justin Matejka, Wei Li, Tovi Grossman, and George Fitzmaurice. 2009. CommunityCommands: Command Recommendations for Software Applications. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology (UIST '09)*. ACM, New York, NY, USA, 193–202. DOI: http://dx.doi.org/10.1145/1622176.1622214

[41] Hugh McLoone, Ken Hinckley, and Edward Cutrell. 2003. Bimanual interaction on the microsoft office keyboard. *Rauterberg, M., Menozzi, M,*

& Wesson, J.(Eds.), Human-Computer Interaction INTERACT'03 (2003), 49–56.

[42] Helena M. Mentis, Kenton O'Hara, Gerardo Gonzalez, Abigail Sellen, Robert Corish, Antonio Criminisi, Rikin Trivedi, and Pierre Theodore. 2015. Voice or Gesture in the Operating Room. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. ACM, New York, NY, USA, 773–780. DOI: http://dx.doi.org/10.1145/2702613.2702963

[43] Kathleen K. Molnar and Marilyn G. Kletke. 1996. The impacts on user performance and satisfaction of a voice-based front-end interface for a standard software tool. *International Journal of Human-Computer Studies* 45, 3 (1996), 287 – 303. DOI: http://dx.doi.org/10.1006/ijhc.1996.0053

[44] Takuya Nishimoto, Nobutoshi Shida, T Koayashi, and Katsuhiko Shirai. 1995. improving human interface drawing tool using speech, mouse and key-board. In *Robot and Human Communication, 1995. RO-MAN'95 TOKYO, Proceedings., 4th IEEE International Workshop on*. IEEE, 107–112.

[45] Donald A. Norman and Stephen W. Draper. 1986. *User Centered System Design; New Perspectives on Human-Computer Interaction*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA.

[46] Uran Oh and Leah Findlater. 2013. The Challenges and Potential of End-user Gesture Customization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 1129–1138. DOI: http://dx.doi.org/10.1145/2470654.2466145

[47] Sharon Oviatt. 1999. Ten Myths of Multimodal Interaction. *Commun. ACM* 42, 11 (Nov. 1999), 74–81. DOI: http://dx.doi.org/10.1145/319382.319398

[48] Randy Pausch and James H Leatherby. 1991. An empirical study: Adding voice input to a graphical editor. In *Journal of the American Voice Input/Output Society*. Citeseer.

[49] Randy Pausch and James H Leatherby. 1991. *Voice input vs. keyboard accelerators: a user study*. Department of Computer Science, School of Engineering and Applied Science, University of Virginia.

[50] Emil Protalinski. 2017. Google's speech recognition technology now has a 4.9% word error rate. https://venturebeat.com/2017/05/17/googles-speech-recognition-technology-now-has-a-4-9-word-error-rate/. (2017). Accessed: 2017-08-11.

[51] Joey Scarr, Andy Cockburn, Carl Gutwin, and Andrea Bunt. 2012. Improving Command Selection with CommandMaps. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 257–266. DOI: http://dx.doi.org/10.1145/2207676.2207713

[52] A. Sears, M. Lin, and A.S. Karimullah. 2002. Speech-based cursor control: understanding the effects of target size, cursor speed, and command selection. *Universal Access in the Information Society* 2, 1 (01 Nov 2002), 30–43. DOI: http://dx.doi.org/10.1007/s10209-002-0034-6

[53] Jana Sedivy and Hilary Johnson. 1999. Supporting creative work tasks: the potential of multimodal tools to support sketching. In *Proceedings of the 3rd conference on Creativity & cognition*. ACM, 42–49.

[54] Vidya Setlur, Sarah E. Battersby, Melanie Tory, Rich Gossweiler, and Angel X. Chang. 2016. Eviza: A Natural Language Interface for Visual Analysis. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 365–377. DOI: http://dx.doi.org/10.1145/2984511.2984588

[55] Ben Shneiderman. 2000. The limits of speech recognition. *Commun. ACM* 43, 9 (2000), 63–65.

[56] Yiwen Sun, Jason Leigh, Andrew E Johnson, and Sangyoon Lee. 2010. Articulate: A Semi-automated Model for Translating Natural Language Queries into Meaningful Visualizations.. In *Smart Graphics*, Vol. 6133. Springer.

[57] Jaime Teevan, Eytan Adar, Rosie Jones, and Michael AS Potts. 2007. Information re-retrieval: repeat queries in Yahoo's logs. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 151–158.

[58] Robert Trevellyan and Dermot P. Browne. 1987. A Self-regulating Adaptive System. In *Proceedings of the SIGCHI/GI Conference on Human Factors in Computing Systems and Graphics Interface (CHI '87)*. ACM, New York, NY, USA, 103–107. DOI: http://dx.doi.org/10.1145/29933.30867

[59] Tom Yeh, Tsung-Hsiang Chang, and Robert C. Miller. 2009. Sikuli: Using GUI Screenshots for Search and Automation. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology (UIST '09)*. ACM, New York, NY, USA, 183–192. DOI: http://dx.doi.org/10.1145/1622176.1622213

[60] Richard M Young. 1983. Surrogates and mappings: Two kinds of conceptual models for interactive devices. *Mental models* 37 (1983), 35–52.

[61] Jingjie Zheng and Daniel Vogel. 2016. Finger-Aware Shortcuts. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 4274–4285. DOI: http://dx.doi.org/10.1145/2858036.2858355

[62] Yu Zhong, T. V. Raman, Casey Burkhardt, Fadi Biadsy, and Jeffrey P. Bigham. 2014. JustSpeak: Enabling Universal Voice Control on Android. In *Proceedings of the 11th Web for All Conference (W4A '14)*. ACM, New York, NY, USA, Article 36, 4 pages. DOI: http://dx.doi.org/10.1145/2596695.2596720