A TRANSITORY INTERFACE COMPONENT FOR THE IN-CONTEXT VISUALIZATION AND ADJUSTMENT OF A VALUE

A Thesis

by

ANDREW WEBB

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2007

Major Subject: Computer Science

A TRANSITORY INTERFACE COMPONENT FOR THE IN-CONTEXT VISUALIZATION AND ADJUSTMENT OF A VALUE

A Thesis

by

ANDREW WEBB

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Andruid Kerne
Committee Members,	Richard Furuta
	Yauger Williams
Head of Department,	Valerie Taylor

August 2007

Major Subject: Computer Science

ABSTRACT

A Transitory Interface Component for the In-Context Visualization and Adjustment of a Value. (August 2007)

Andrew Webb, B.S., Texas A&M University

Chair of Advisory Committee: Dr. Andruid Kerne

Some agent-based systems depend on eliciting ratings from the user. However, the user's willingness to provide ratings is limited due to requisite demands of attention and effort. From a human-centered view, we redefine providing ratings as expressing interest. We develop a new interface component for parameter setting, the *In-Context Slider*, which reduces physical effort and demand on attention by using fluid mouse gestures and in-context interaction. We hypothesize that such an interface should make interest expression easier for the user.

We evaluated the In-Context Slider as an interest expression component compared with a more typical interface. Participants performed faster with the In-Context Slider. They found it easier to use and more natural for expressing interest. We then integrated the In-Context Slider in the agent-based system, combinFormation. We compared the In-Context Slider with combinFormation's previous interest expression interface. Of the participants that effectively used both interfaces, most expressed more interest with the In-Context Slider. Participants' experience reports described the In-Context Slider as easier to use while developing collections to answer open-ended information discovery questions.

This research is relevant for many applications in which users provide ratings, such as recommender systems, as well as for others in which values need to be adjusted on many objects that are concurrently displayed.

ACKNOWLEDGMENTS

I am endlessly beholden to Andruid Kerne, for his continuous encouragement, commitment, and direction. I have gained irreplaceable skills and experiences from his guidance. He has taken me through a beautifully enthralling, yet trying, educational endeavor that has built a diverse and firm foundation for my future work.

A special thanks to Yauger Williams for his enduring engagement and advice in my research process.

I would like to thank Richard Furuta for his insight and involvement.

I express my immense gratefulness to all the members of the Interface Ecology Lab for their help, suggestions, and support. I would especially like to thank Eunyee Koh, Ross Graeber, Zach Toups, Megan Schneider, Nathan Robinson, and Blake Dworaczyk for all the assistance and support provided.

Finally and most importantly, to my parents, I could have never accomplished this without you. Your unconditional love and support have given me strength in the most waning of times.

TABLE OF CONTENTS

Page

ABSTRACT iii

ACKNOWLEDGMENTS v

TABLE OF CONTENTS vi

LIST OF FIGURES viii

LIST OF TABLES xi

1 INTRODUCTION 1

2 IN-CONTEXT INTERFACE DESIGN ISSUES 3

- Attention 3
- Limited Screen Real Estate 5
- Activation Issues 6
- Recognizing Context 8

3 RELATED WORK 9

- Eliciting Ratings from Users in Recommender Systems 9
- Fluid and Contextual Interfaces 10
- Prospective Memory 15

4 THE IN-CONTEXT SLIDER 16

- Layers of Activation 16
- Visualizing Values 19
- Interacting to Change a Value 21
- Multi-activation 21
- Implementation 24

5 EVALUATION – EXPERIMENT 1 25

- Participants 25
- Method 25
- Results - Quantitative 30
- Results - Qualitative 33
- Discussion 34

Page

6 INTEGRATING INTEREST EXPRESSION WITH AUTHORING....36

Prior Double Modal Toolbar Interface for Interest Expression and Authoring....39
In-Context Interface for Interest Expression and Authoring....41
combinFormation Layer 0 In-Context Slider Activators....43

7 EVALUATION – EXPERIMENT 2....51

Participants....51
Method....51
Results - Quantitative....54
Results - Qualitative....58
Discussion....59

8 CONCLUSION....62

REFERENCES....64

APPENDIX A....68

APPENDIX B....75

VITA....77

LIST OF FIGURES

Page

Figure 1: Popup vs. in-context interfaces.....4

Figure 2: Examples of layer 0 activators; (a) text, (b) image, (c) passage of text.....17

Figure 3: Examples of layer 1 activators: (a) full circle navel, (b) half circle navel for text objects. 18

Figure 4: Examples with all three layers of activation: (a) text, (b) image, (c) passage of text 19

Figure 5: Visualizing In-Context Slider value: (a) collapsed positive value (b) expanded positive value, (c) collapsed negative value, (d) expanded negative value. 20

Figure 6: In-Context Slider multi-activation sequence.....23

Figure 7: Rating images of cars using Typical Slider Dialog Box interface. 27

Figure 8: Rating images of cars using In-Context Slider.....27

Figure 9: Rating words with the In-Context Slider. 28

Figure 10: Rating words with the Typical Dialog Box Slider interface. 29

Figure 11: Time performance: which interface were participants faster with. 31

Figure 12: Time performance: average times to complete tasks.....31

Figure 13: Participants' experience reports: which interface was easier to use?.....32

Figure 14: Participants' experience reports: which interface was more natural for expressing interest?.....33

Figure 15: Feedback loop of agent and human information processing.....37

Page

Figure 16: combinFormation Double Modal Toolbar interface. Left: Design Toolbar; right: Interest Expression Toolbar. 38

Figure 17: Positioning of details-on-demand, in-context tools, edit palette, and Surrogate In-Context slider for Double Modal Toolbar interface..... 40

Figure 18: In-Context Interface components (clockwise from top): details-on-demand, edit palette, tools, and Surrogate In-Context slider navel. 41

Figure 19: combinFormation surrogates (a) text surrogate, (b) two overlapping image surrogates, (c) fully activated text surrogate, (d) one of two overlapping image surrogates is fully activated..... 44

Figure 20: Crossed out navel of In-Context Slider for stop words..... 46

Figure 21: Text activated In-Context Slider changing the color of the text to reflect the current value. 47

Figure 22: Text surrogate with In-Context Slider adjusting interest in a word and changing the color of related terms within the text surrogate. 47

Figure 23: Example of activated In-Context Slider for a word in details-on-demand field. 48

Figure 24: Example of activated In-Context Slider for details-on-demand field label. ... 50

Figure 25: For participants who expressed interest: which interface was used more for interest expression? Also, which interface was easier to use for expressing interest?..... 53

Figure 26: Expressions of interest. For participants who expressed interest, how many times did they change the interest level with each interface?..... 55

Page

Figure 27: Composition of surrogates created by a study participant for the study abroad information discovery question. An In-Context Slider can be activated for each surrogate and each word. 59

LIST OF TABLES

Page

Table 1: Emergence and quality metrics used for rating participants' answers to information discovery tasks..... 56

Table 2: Ecologylab package classes and their corresponding implementations or subclasses in combinFomation..... 71

1 INTRODUCTION

Some agent-based systems rely on the user to provide ratings on relevant information. These ratings serve as the basis for semantic models that the agents use to make decisions. Prior systems have found that eliciting the user's input on ratings is sufficiently difficult that it proves to be a barrier of entry to these systems actually being used to complete tasks [Balabanovic and Shoham 1997; McNee et al. 2003]. We hypothesize that the design of interfaces for rating recommendations for relevance can play a key role in user adoption of such systems. For example, an online movie rental service provides recommendations to its customers, but first those customers must express what movies they enjoy. This normally requires applying ratings to many different movies. However, the user is often unwilling to engage the ratings interface as it demands more time and attention than is desired.

While most research in information visualization techniques involves providing access to and understanding of high dimensional data, the present research is concerned with contextualized visualization and adjustment of a one-dimensional value. This seemingly pedestrian issue plays an important role in the usability of recommender systems and other tools that require parameter setting for many concurrently displayed objects.

We redefine providing ratings in a human-centered way, as "expressing interest." We develop a fluid in-context interface for interest expression, which can be tightly integrated into other user tasks, such as authoring and editing of textual and visual information. We present a new interface component for the visualization and adjustment

---

This thesis follows the style of the *Journal of the ACM*.

of a value at the point of focus. Our goal is to encourage expressive interaction by reducing user effort and increasing feedback and expressivity.

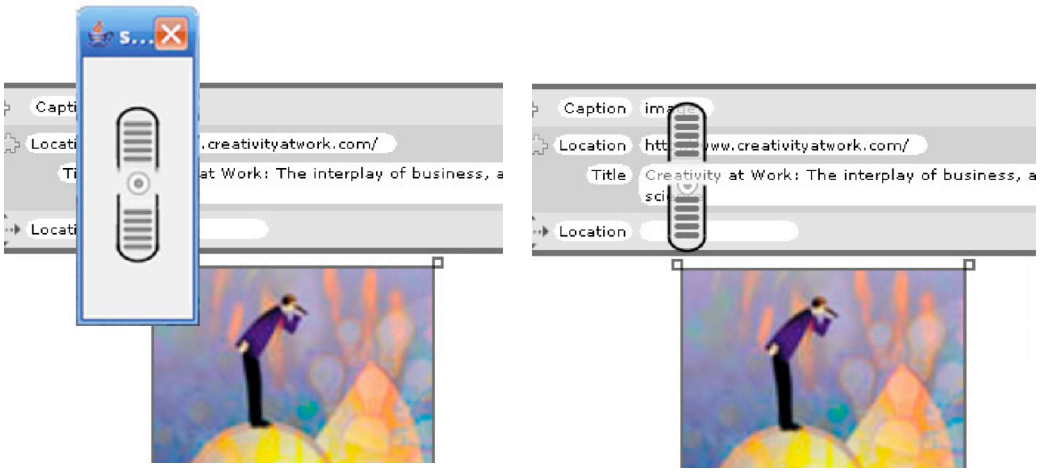
In this thesis, we first explain the issues of designing an in-context interface. Next, we review related work. We then develop the In-Context Slider and an evaluation. We proceed to introduce the integration of interest expression with authoring, and describe how the In-Context Slider affords this in combinFormation [Kerne et al. 2006]. Continuing, we present an evaluation of the In-Context Slider for expressing interest to represent collections with composition. We conclude by deriving implications of this research.

2 IN-CONTEXT INTERFACE DESIGN ISSUES

Interactive spaces often contain large numbers of objects. Users need to perform complex operations on these objects. Hard problems for interface designers include the limits of human attention, the limits of available screen real estate, methods for activation of interface components, and recognizing context of the user's situated task. In-context interface design addresses these issues by providing affordances in-place, making their activation *transitory*, that is, only appearing when necessary and requested, developing clear mappings based on fluid gestures, and basing activation rules on the user's current action.

Attention

Our attention as human beings is limited by the ephemerality of our short-term memory. Our eyes are constantly receiving images, which if not instilled in long-term memory are quickly discarded. Visual working memory is the cognitive mechanism we use for handling visual information [Baddeley 1992; Ware 2004]. The capacity of visual working memory is restricted to only a few, simple objects (three to five). From a user interface standpoint, the more visual objects that are required to complete a task, the harder it is for the user to maintain focus as some of the objects may be removed from the visual working memory to make room for new ones. This restriction on memory results in short attention spans. If a person can only maintain three to five visual objects but is constantly bombarded with new visual stimuli, it becomes incredibly difficult for a person to maintain attention especially for tasks involving many steps.



(a) (b)

Figure 1: Popup vs. in-context interfaces.

Shneiderman and Bederson proposed three strategies to help better maintain user attention: reduce short-term and working memory load, provide information abundant interfaces, and increase automaticity [Shneiderman and Bederson 2005]. Automaticity is the ability to perform complex actions with minimal cognitive load, an aspect common among experts of a particular system. For reducing memory load, they suggested using an effective display design that conveys decision-important information readily or in-context. By providing information abundant interfaces (without exceeding the necessary amount of information), the short-term memory load for the user is reduced as the required information is displayed on screen. Increasing automaticity helps the user quickly execute common commands while requiring a minimal cognitive effort (e.g. keyboard shortcuts for programs like Photoshop).

In-context interfaces work to alleviate the problems of limited attention by displaying information in-place, preventing saccadic eye movements (quick, jerky) and

limiting the amount of new visual objects brought into the visual working memory. Figure 1 shows the difference between a slider in a popup window (a) and a slider in-context (b). The popup window not only adds more visual objects, but obscures those already on screen; whereas, the in-context visualization only adds the necessary visual objects while minimizing the amount of overlapping.

Another means of dealing with the limits of attention and memory involves designing interface components that use fluid motion, reducing the physical effort required of the user. For example, rather than require the user to click and drag a knob in a slider, instead we simply use directional mouse movements over the slider to adjust the value (knob position). This idea of fluid motion raises design issues regarding activation and context. A user could accidentally adjust the value of a fluid slider by moving the mouse over the slider in the proper motion if the slider were always visible and always activated for adjustment. These issues of activation and context are discussed later on in this section.

Limited Screen Real Estate

The limits of the screen space are problems faced by all user interface designs. Screen real estate is valuable and must be used efficiently; otherwise the user's attention is wasted. For example, a screen space overly cluttered with interface components requires extra effort on the part of the user to decipher for each task where the appropriate interface component(s) is located.

In the case of in-context interface components, screen space is even more limited since interaction needs to occur in proximity to an already present interactive object. In many cases the focus component is also surrounded by other components. As a result of

these space constraints, in-context interface components are transitory. Depending on the instance, in-context interface activation can either change the layout of other objects [e.g. Zellweger et al. 1998] or overlay those other objects. Problems occur with each of these. Changing the layout requires the user to cognitively process the movement of objects. Overlaying can cause other objects to become obscured and inaccessible. For this reason, a design goal is to make in-context interface components utilize minimal screen real estate. Making the components transitory also means that special forms of activation are necessary since the component may have no persistent visible representation on the screen.

Activation Issues

Fitts' law states that the time needed to acquire a target is the function of the distance to the target and the target size [Fitts 1954; MacKenzie and Buxton 1992]. Therefore, the further a person has to move a mouse pointer in a task, the longer it takes that person to both find where to move the mouse pointer and to move the pointer to the target. In order to increase efficiency and accuracy, targets for a task need to be placed within as close a range as possible of each other. Ideally, targets appear directly in a person's current area of focus, avoiding the need for switching context from the task at hand to find the target required to complete the task.

The placement of targets is not a simple task, as visual space is limited and consistent mappings of controls must be maintained. For example, a focus object in the interactive space may have several different actions that can be performed on it. The focus is small in size, e.g., a word on this page. The focus is surrounded in close proximity by several similar objects. The targets for the possible action cannot simply be

placed around the focus object as some of these targets may overlap the other neighboring objects. It becomes evident that these targets need to be transitory. Making the targets transitory helps solve the placement issue by hiding targets when not required.

Activation issues result from problems with limited screen space and the use of fluid motions. Special activation methods are needed to address problems with multiple components residing in the same location. To avoid confusion for the user, these activation methods must be designed carefully, so as to avoid unclear mappings and affordances [Norman 1988]. For example, if an interface component requires the user to click check boxes to determine which objects a command should affect, but the check boxes look like non-clickable decoration, the user would have difficulty knowing where to click to select objects. Maintaining clear mappings and affordances grows increasingly difficult as the density of interface components increases.

Common solutions to activation issues include keyboard shortcuts and the use of right click popup menus. Right click popup menus, although appearing in-context, require the user to switch focus from the task to deciding which action from the popup menu is necessary. Keyboard shortcuts are efficient methods for activation, but they require the user to memorize a list of keyboard commands that aren't obvious at first use. The user is required to recall commands from memory rather than recognize them from affordances. For the standard mouse-keyboard configuration used by most computers today, if the keyboard is excluded from activation methods, then we are left with only the mouse with two buttons (one button for users of many Apple computers). These limitations prefigure the difficulties in designing appropriate activation methods for in-context transitory interface components.

Recognizing Context

Of course, developing in-context interface components requires that the designer recognize some sense of the user's situated task context. A transitory interface, in particular, must have some sense of the user's intentions, in order to know when it needs to be visible; visibility can be minimized or eliminated at other times. The better a component recognizes the user's context, the better able it is to appear when necessary. Portions of an in-context interface may be activated in contextual layers. Recognizing context is by no means a simple undertaking, especially as the number of possible actions increases.

3 RELATED WORK

This research is related to prior work regarding recommender systems and fluid interfaces. As an interest expression mechanism, the In-Context Slider builds on prior research on ratings in recommender systems. We extend prior work in fluid and contextual interface design. We briefly examine research on prospective memory which is relevant to designing interfaces that reduce cognitive effort.

Eliciting Ratings from Users in Recommender Systems

Before describing the relation of recommender systems to this research, it is important to note that recommender systems are only one type of system that could benefit from the In-Context Slider. The In-Context Slider was designed as interest expression mechanism, a new tool for "providing ratings." However, the functionality of the In-Context Slider is not limited to these systems. Other systems, where the user is required to provide a value through a slider interface, could improve user experience and efficiency through use of the In-Context Slider.

Recommender systems are agent-based tools that work to find documents relevant to a user's interests. Providing ratings is a quintessential component of these systems. Recommender systems use the ratings, and techniques such as collaborative filtering [McNee et al. 2003] and information retrieval models [Baeza-Yates and Ribeiro-Neto 1999] to make choices about what information resources from a larger collection to retrieve for a user. Providing ratings is personal and contemplative, requiring focus and attention. The process necessitates that the user make decisions about how interesting things are. The user must assign a valence, a positive or negative value, regarding relevance.

Despite the benefits of interest expression to the user, the extra effort required discourages users from rating recommendations. McNee et al. [2003] researched differences between user-controlled and system-controlled recommender systems. By user-controlled, they mean a system in which the user decides when to make recommendations. They discovered that while the user-controlled system increased user burden, this system also provided users with more relevant results. While the user-controlled system required the most amount of time to use, some users did not seem to notice, due to a sense of increased engagement. However, the greater effort required by the user-controlled system resulted in fewer users completing the assigned tasks.

Others describe similar problems with getting users to provide ratings [Balabanovic and Shoham 1997, Ha and Haddawy 1998]. Fab is a hybrid recommendation system using two types of recommendation methods as a way to obtain equivalent or better results with fewer ratings required by the user [Balabanovic and Shoham 1997]. Ha et al. propose using an interface that creates a default preference representation for new users based on the next closest pre-existing representation based on other users in order to reduce the number of preference elicitations from a user [Ha and Haddawy 1998].

Fluid and Contextual Interfaces

Over past few years, several different interface components have been developed involving the use of in-context visualizations and space constrained sliders.

FlowMenu is a marking menu designed for a display surface with a pen input device and allows for in-context execution of commands by making gestures with the pen device [Guimbretiere and Winograd 2000]. FlowMenu applies several of the same

interaction principals designed for the In-Context Slider. FlowMenu uses motions that are natural and intuitive to the user to improve performance.

FaST sliders combine marking menus [Kurtenback and Buxton 1993] and the typical slider to create a new slider interface component with three stages [McGuffin et al. 2002]. In the first stage, a marking menu [e.g. Guimbretiere and Winograd 2000] selects the value to be adjusted. The second stage adjusts the value. The third stage allows the use of additional controls to affect the value. Removing the first stage, FaST sliders and the In-Context Slider are similar. However, the FaST slider requires the user first position the slider and then adjust the value using an extra mouse drag event. This mouse drag event, as noted by the authors, can lead to setting the wrong value if the user moves the mouse while ending drag or releasing the mouse button too soon.

Fluid links are a mechanism for hypertext created by Zellweger, et al. where information about a hyperlink is displayed in-context to better help the user in deciding which hyperlinks to follow [Zellweger 1998]. When a user mouses over a fluid link, the visual layout of the hypertext document is modified by the addition of new information about the link (such as the first few lines of the linked page) placed on the line below (moving all lines below down a few lines) the link or in a margin to the right or left of the fluid link. Fluid links are similar to the proposed in-context interface in that a layer of activation is used when the user mouses over a fluid link.

Side Views is a user interface component that provides on-demand details along with persistent and dynamic previews for a given command [Terry and Mynatt 2002]. Side Views supports open-ended tasks in which case it is unclear the sequence of steps required to reach the desired final solution. Side Views was implemented in the GNU

Image Manipulation Program (the GIMP) [GNU 2007] and provides in-context visualization by displaying previews directly next to the point a command is selected and executed (e.g. a menu item from a drop-down menu). While Side Views are transitory by default, they can be made persistent if the user desires. In a persistent form, the Side Views window remains on screen until the user closes it, and can still be used to make changes or moved to a different location.

Local Tools, developed by Bederson et al., is an alternative to tool palettes and arguably the antithesis of the In-Context Slider [Bederson et al. 1996]. Local Tools provides the user with tools that can be picked up, used, and then dropped anywhere on the screen. This idea differs from the standard tool palette in that tools are fixed to single location. The user can place different tools in different places in hopes to improve interaction efficiency by allowing tools to be located and/or moved closer to the point of command execution. Local Tools still suffers from the same problem as the standard tool palette in that the user must still shift focus to select the tool.

Created by Stephen Eick, the Data Visualization Sliders use information visualization techniques to enhance sliders [Eick 1994]. Data Visualization Sliders use the screen real estate used by the sliders to visualize information in the form of graphs with both continuous and discrete values. The graphs show information related to the data represented by the slider to help the user in selecting a value for the slider.

Koike, et al. [1997] created the TimeSlider built on ideas from Eick's Data Visualization Sliders as a means for adjusting a value that has a large range. The TimeSlider implemented as a history selection mechanism displays time values in a non-linear manner. In the middle portion of the slider is a set of linear values with a finer

grain of adjustment, while in the top and bottom sections are non-linear values with coarser adjustments. In order to make more accurate selection possible in the coarse sections, up and down arrow buttons are provided to allow movement of values into the middle finer adjustment section. The TimeSlider provides a technique for use in any slider using large varying ranges.

Tsandilas and Schraefel [2003] present a system and interface for allowing users to express interest in topics, and based on the interest expression, web pages are rendered using visual design and information visualization techniques to help emphasize information on the pages that is of interest to user. In the interface for this system, sliders are used to allow users to express interest in topics, although the sliders and interface are in a separate window not used in-context.

See-Through tools are translucent tools located on a plane above the interactive space [Bier et al. 1994]. The user interacts with objects through these tools applying the tools' effects to the objects below. The tools can be moved around the screen, between applications, and layered on top of each other. While the In-Context Slider is not a See-Through tool; it shares the translucence quality, and the layers of activation. Although serving different functional roles, they are similar in concept to the layering capabilities of See-Through tools.

Henderson and Card [Henderson and Card 1986] designed a window manager that uses window access statistics and multiple workspaces to solve the problems associated with limited screen space. They described a problem known as "window thrashing" where a large amount of effort on the part of the user is required to keep the desired information visible on screen (constantly switching between the various windows

on the screen). Their solution involves creating multiple virtual workspaces called *Rooms* such that each represents one of the various categories of workspaces (e.g. mail, office, programming). The system provides a pop-up for selecting between rooms. Entering one of these Rooms opens pre-arranged and pre-sized application windows. By doing this, they lessen the amount of "window thrashing" for the user by reducing the overlapping of windows.

McGuffin and Balakrishnan [2002] studied the user performance effects of having an interface component that expands based on the interest and focus of the user. They ran an experiment where a set of subjects using a computer with a mouse were asked several times to acquire a target which in some cases was expanding as the mouse moved closer and in other cases remained the same size. Their experiment found that user performance was consistently improved when using expanding targets and that these improvements were dependant on a target's final size as opposed to its initial size.

Baecker, et al. [1991] designed a form of animated icon that provides an in-context visualization of the action represented by the icon in hopes of providing better affordances to the user and therefore improving user performance. They implemented these animated icons in a tool palette for a drawing program. When the user rolled the mouse over one of these icons, the icon would animate a short sequence illustrating the purpose of the tool. Baecker conducted a user study in which users found the animated icons helpful when the purpose of a tool was unknown.

Prospective Memory

Prospective memory is memory related to the cognitive effort of remembering future intended actions. In recent years an increasing amount of research has gone into understanding prospective memory [Sellen et al. 1997].

Sellen, et al. [1997] conducted a study on prospective memory in the work place. They asked participants to perform a time task one week and a place task the next. The tasks involved triple-clicking a button on a badge twice at certain time intervals for the time task and when entering certain places for the place task. They also asked the participants to triple-click once whenever they thought about one of these tasks (no matter where they were located). Results from the study showed that the participants responded more accurately to the place task, but thought more often about the time task. Thus, the researchers determined that cues are necessary to help in prospective remembering, while without cues (in the case of the time task) extra cognitive effort is needed to remember tasks. Although not directly related to the cognitive effort involved in using interface components, this research does point out the need for visual cues to help reduce the cognitive load on the user.

4 THE IN-CONTEXT SLIDER

The In-Context Slider is a user interface component that recognizes aspects of the user's situated task to provide transitory affordances in proximity to the focus object to support the adjustment of a value through fluid movements. We arrived at this solution through a human-centered iterative design process.

Layers of Activation

What makes an in-context interface fluid is the ability to activate layers of interface at the point of focus, in the midst of an interactive space, through simple gestures. Clear affordances are required to cue the user about how to trigger each successive layer. We call these affordances *activators*. An activator provides fluid transitions between the layers of interaction. Activation affordances must be designed so that their presence minimally disrupts other constituent functionalities of the context. The affordance for each successive layer of activation is positioned in-context, relative to the positions of the preceding activators. In order to prevent unwanted activations, a delay may be necessary before visualizing each layered activator.

An In-Context Slider has three layers of activation. Each layer is activated by the mouse-over gesture. The activator in the initial layer, layer 0, is an already present object in the interactive space with pre-existing functionality, which can be augmented by an In-Context Slider. As an activator, this object receives new functionality as an affordance for accessing the next layer of activation. In the present research, a layer 0 activator can be an image, a word in a passage of text, or a whole passage of text (see Figure 2a, b, c). The functional contribution of an activator does not disrupt other functionality. Thus, text that is editable remains editable, while each word may be augmented to enable interest

expression. The location for the layer 1 activator is one that places it in close proximity to layer 0, while avoiding the occlusion of visual features that are otherwise necessary for the legibility and usability of the context. Since a layer 0 activator has additional pre-existing functionality, mousing over it does not necessarily mean the user desires to activate an In-Context Slider. The user could be simply passing over the activator to interact with something else. To handle this issue, a small adjustable delay (defaulted to 550ms) is applied before visualizing the level 1 activator. Interaction with the pre-existing functionality of a layer 0 activator, such as clicking to type a character amidst text, or click and drag to highlight, results in the immediate removal of a layer 1 activator. Pulling the mouse off the layer 0 activator and not onto the layer 1 activator, also removes a layer 1 activator.

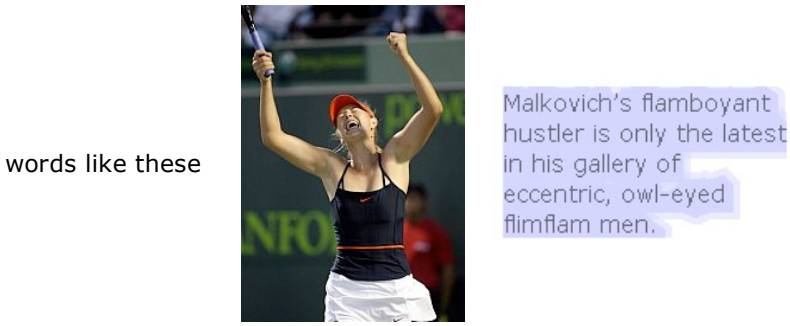


Figure 2: Examples of layer 0 activators; (a) text, (b) image, (c) passage of text.

In the In-Context Slider, the layer 1 activator is an affordance called the *navel*. The navel is a small circular object that is designed to be differentiable from, yet not disruptive of its surroundings (see Figure 3), and to form the center of the subsequent layer 2 In-Context Slider body (see Figure 4). The navel comes in two different visual

forms to accommodate the variety of layer 0 activators that activate it. For images and surrogates within combinFormation, the navel is a full circle (see Figure 3a). For text, the navel is the bottom half of the full circle version (see Figure 3b). The horizontal edge forming the top of the half circle navel fits visually with the base line of text. As well, text is normally formed by a horizontal sequence of words across vertical arrangements of lines. The gap between the lines provides an appropriate unused space to place the navel. In combinFormation, text surrogates are editable pieces of text where a user may wish to express interest on the words inside a text surrogate. To avoid interaction complications between text editing and activation of navels for words inside a text surrogate, the navel for surrogates is placed directly to the left side of a surrogate (see Figure 4c, d).

Figure 3: Examples of layer 1 activators: (a) full circle navel, (b) half circle navel for text objects.

Layer 2 is visualized by the body of the In-Context Slider, which expands vertically outward from the navel. The slider body contains a set of vertically stacked horizontal bars representing the possible values for the slider. The horizontal bars are split across the navel, so that bars representing positive values appear above the navel and bars representing negative values appear below the navel (see Figure 4). The total number of bars can be adjusted. The default number is ten, five positive and five negative. A slight translucency is applied to the slider body in the area surrounding the bars. This

translucency allows visual objects possibly occluded by the slider body to still remain partially visible. As an in-context interface designed to minimize the cognitive effort on the user, keeping the focal point of the interactive space optimally visible is an important task. The translucency also gives the slider body a lighter than air quality, which is representative of its transitory nature as a layer of activation. Mousing off the slider body but onto the layer 0 activator removes the slider body and leaves the navel. Mousing off the slider body and off the layer 0 activator in the process removes both the slider body and the navel.

(a) (b) (c)

Figure 4: Examples with all three layers of activation: (a) text, (b) image, (c) passage of text.

Visualizing Values

The present research applies Norman's prescription, to "make things visible" [Norman 1988]. The current value of an In-Context Slider is visualized by highlighting, with hue, the bars in the slider body that represent the value (see Figure 5). Color is a pre-attentive visual feature [Nagy and Sanchez 1990]. In our vision, hue is processed early and in parallel requiring no attention. This cognitive property of color makes it well-

suited for visualizing value in an In-Context Slider. With the In-Context Slider body, positive values are represented in green. Negative values are represented in red. The neutral value is represented by gray. Since gray is an entirely unsaturated color, the saturation of the color is used to represent the intensity (distance from zero) of the value. In other words, a positive value of five has a much higher saturation than a positive value of one. A value of five will appear greener than a value of one. The same applies to negative values with the color red. To handle physiological (e.g. color blindness) and cultural issues, the hues for positive and negative can be changed. Green and red are the default.

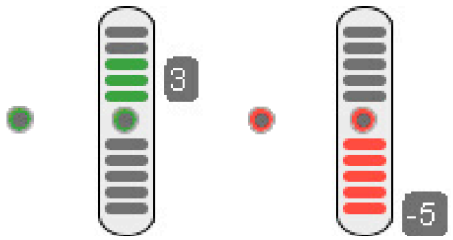


Figure 5: Visualizing In-Context Slider value: (a) collapsed positive value (b) expanded positive value, (c) collapsed negative value, (d) expanded negative value.

The navel and activator provide mechanisms for visualizing the value of an In-Context Slider even when the slider is not activated to the third level. Inside the navel is a light gray ring that changes color to match the current value (see Figure 5). This allows the In-Context Slider, while not fully expanded, to visualize whether the current value is positive, negative, or neutral and provide some indication of the intensity of that value (see Figure 5a, c). The level 0 activator of an In-Context slider can also have its

appearance adjusted to reflect the current value. For example if an activator is a textual word, the color of the word will change to match the color for its assigned value. This provides quick feedback to the user about the currently assigned value. In combinFormation, this is the level of interest expression.

Interacting to Change a Value

To change the value of an In-Context Slider, the user moves the mouse cursor up or down over the layer 2 slider body. All bars from the navel (center) to the current mouse position are highlighted with the appropriate color (see Figure 5). A small popup textbox with the current visualized value appears to the side of the slider vertically matching the current mouse position. Once the desired value is visualized, the user clicks the left mouse button to set the value, and, depending on whether the mouse cursor is currently over the activator or not, the In-Context Slider either reverts to the collapsed navel-only form or disappears entirely. The user can choose not to change the value by simply moving the mouse off the In-Context Slider without clicking. If, after moving the mouse off, the mouse cursor is still positioned over the layer 0 activator, the In-Context Slider layer 1 remains in collapsed navel-only form. If the mouse cursor ends off the layer 0 activator, the In-Context Slider is fully deactivated, removing it entirely (both layer 1 and layer 2) from the screen.

Multi-activation

In the iterative design process, it was discovered that a user may wish to set the same value to multiple objects at one time. To accommodate this action, multiple layer 0 objects can be activated at once. Layer 1 navels remain visible for each activated layer 0 object through the course of the activation sequence. The process of multi-activate is

similar to that of marking a route on a map through a set of waypoints. The waypoints are the navels of the In-Context Sliders (see Figure 6). The user enacts multi-activate by holding the left mouse button down while over the navel and dragging the mouse cursor. A fuchsia-colored line is drawn from the center of the navel to the current mouse cursor position. While dragging, the user can mouse over another layer 0 activator, causing another navel to appear. In this case, the delay for showing the navel is removed since the intention to activate additional In-Context Sliders is clear from context. If the user ends drag by releasing the left mouse button while over the new navel, the fuchsia line disappears and a persistent gray line is drawn connecting the center of the two navels, just as a connecting line marks a route segment between two waypoints on a map. Since the user is now over a navel, the slider body is activated. The user can continue activating In-Context Sliders by repeating the same process from the current navel to another navel. After activating the desired sliders, the user changes the value of the last activated slider. This changes the value for all other activated sliders. Multi-activation is cleared when the user either sets a value or deactivates an activated In-Context Slider.

Figure 6: In-Context Slider multi-activation sequence.

When activating multiple sliders, it is not required to end the mouse drag on a navel. If the mouse drag is ended on the layer 0 activator, the slider connected with that activator will be activated, drawing a gray line between the navels. Multiple-activation

doesn't have to start at the navel. It can also start from the slider body. The process is the same as when starting from the navel (hold left mouse button and drag). The difference is that when activating another slider (by ending drag), the current value for the newly activated slider is set to the value of the previously activated slider. In other words, by starting multi-activation in a slider body, the current value is propagated to each slider activated afterwards in the activation sequence. This multi-activation sequence provides flexibility in assigning the same value of interest to multiple objects. If at any point in the process the user decides a different value is appropriate, that value can easily be assigned from the current slider, and the sequence can continue.

Implementation

Details about the implementation of the In-Context Slider and how application developers can use the In-Context Slider in their applications can be found in Appendix A.

5 EVALUATION – EXPERIMENT 1

The goal when developing the In-Context Slider was to build a fluid, in-context interface component that improves expressivity by reducing user effort. Experiment 1 was designed to measure the easy of use of the In-Context Slider in comparison to a more typical interface for interest expression.

Participants

Forty-three student volunteers participated in the experiment. Undergraduate members of the "psychology subjects pool" fulfilled a requirement of their introductory psychology course by participating. Concurrently offered sections of the course had a total enrollment of more than 1000 students. The subjects represent a spectrum of undergraduates, with no focus on computer or information science majors. The experimenters were not personally familiar with the participants.

Method

Two tasks were designed to evaluate the In-Context Slider in comparison to a Typical Dialog Box Slider interface for interest expression. The Typical Dialog Box Slider represents a common interface for making adjustments to a value for an object. The Typical Dialog Box Slider interface consisted of a drag-able slider with a knob inside a dialog box with OK and Cancel buttons. The dialog box was activated through a right-click popup menu. The popup menu contains several options (e.g. copy, cut, paste) including one labeled, "Set the Interest." The addition of other popup menu options simulates the real world situations where the right click popup provides extra functionality. Before completing each task, an instructional video was shown explaining

the task and how to use each interface to complete it. Participants were given a brief practice session before using both interfaces.

In Task 1, participants were asked to rate a collection of images of automobiles according to their personal taste, using the two different interfaces, the In-Context Slider and the Typical Dialog Box Slider. Images were displayed four at a time, each labeled above with a single letter. To match the In-Context Slider's ability to assign the same value to multiple objects, multiple images in the traditional slider interface were selectable using a series of CTRL-clicks (holding CTRL key while clicking the left mouse button). Clicking the right mouse button on any selected image and selecting "Set Interest" from the popup menu brings up the dialog box with each selected images' label comma separated and printed below the slider (see Figure 7). For the In-Context Slider interface, an In-Context Slider was placed in the center of each image (see Figure 8). Multiple images could be rated at once using the In-Context Slider multiple selection mechanism.

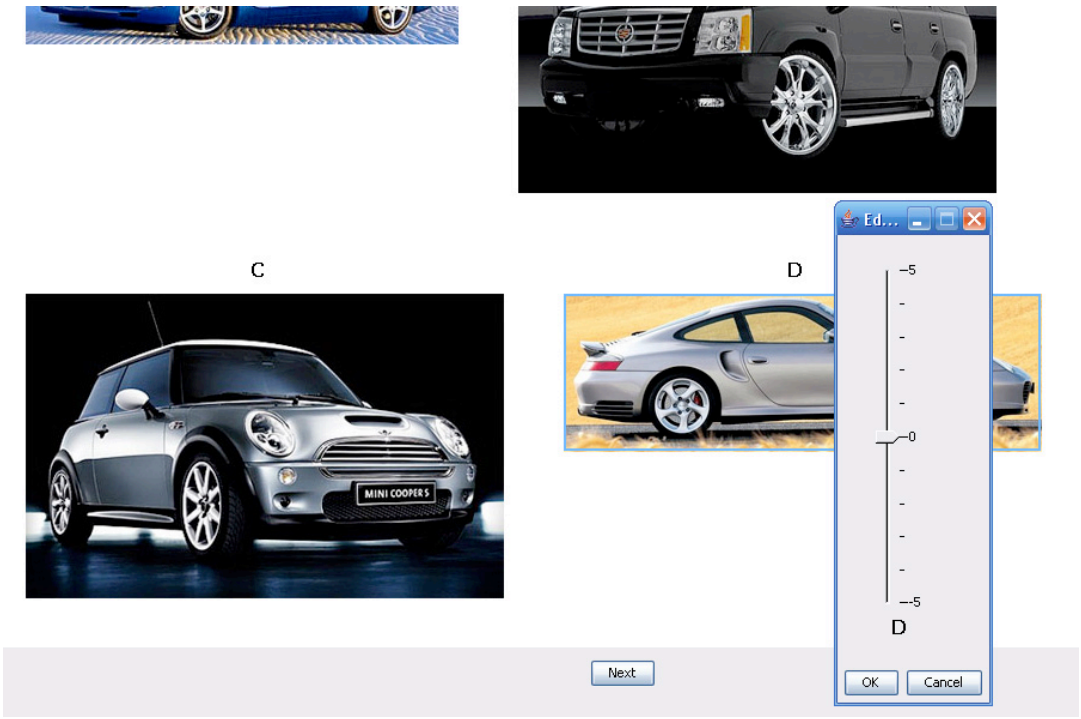


Figure 7: Rating images of cars using Typical Slider Dialog Box interface.

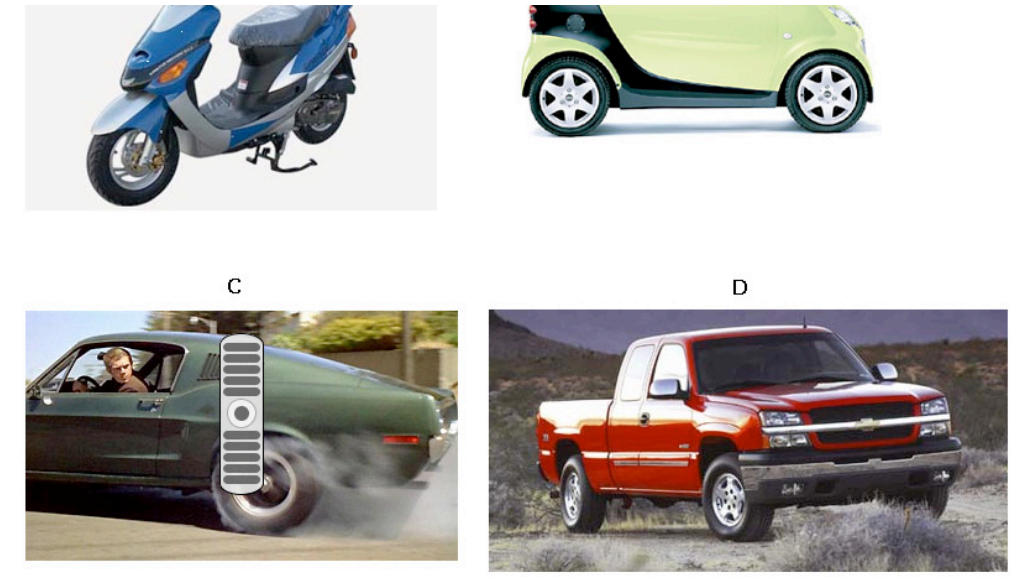


Figure 8: Rating images of cars using In-Context Slider.

Task 2 was different from Task 1 in that rather than rating images, participants were asked to rate single words in a passage of text (see Figure 9,10). The context was as if one was editing the text, and wished to express interest in particular words in the context of the editing task. However, editing was not in fact part of the task in this simulation. The two rating interfaces, the In-Context Slider and the Typical Dialog Box Slider were the same as before. The layer 0 activators were words, instead of images. Further, in this task, instead of spontaneously and personally rating words, participants were provided with a value to assign to each word. This value was located in the text, in parentheses, following the word, to maintain contextual continuity in the participant experience. Words that required rating were presented in bold face to distinguish them from the other words.

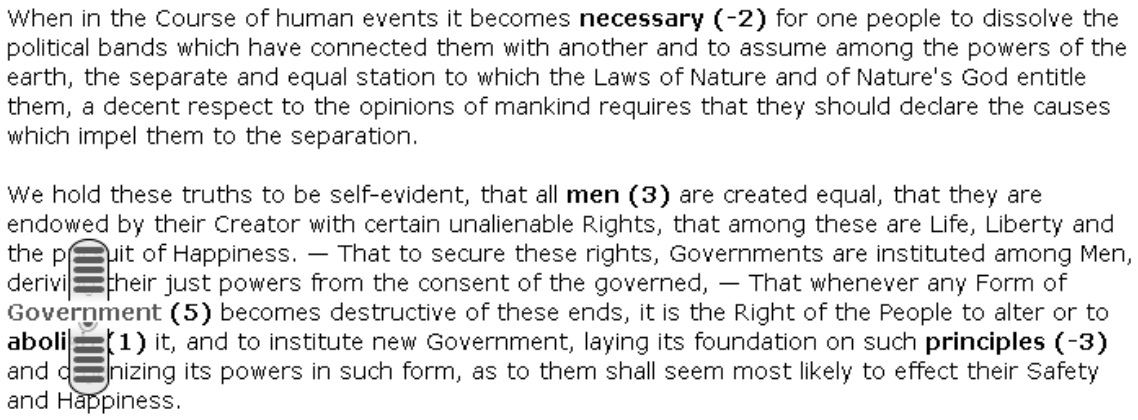


Figure 9: Rating words with the In-Context Slider.

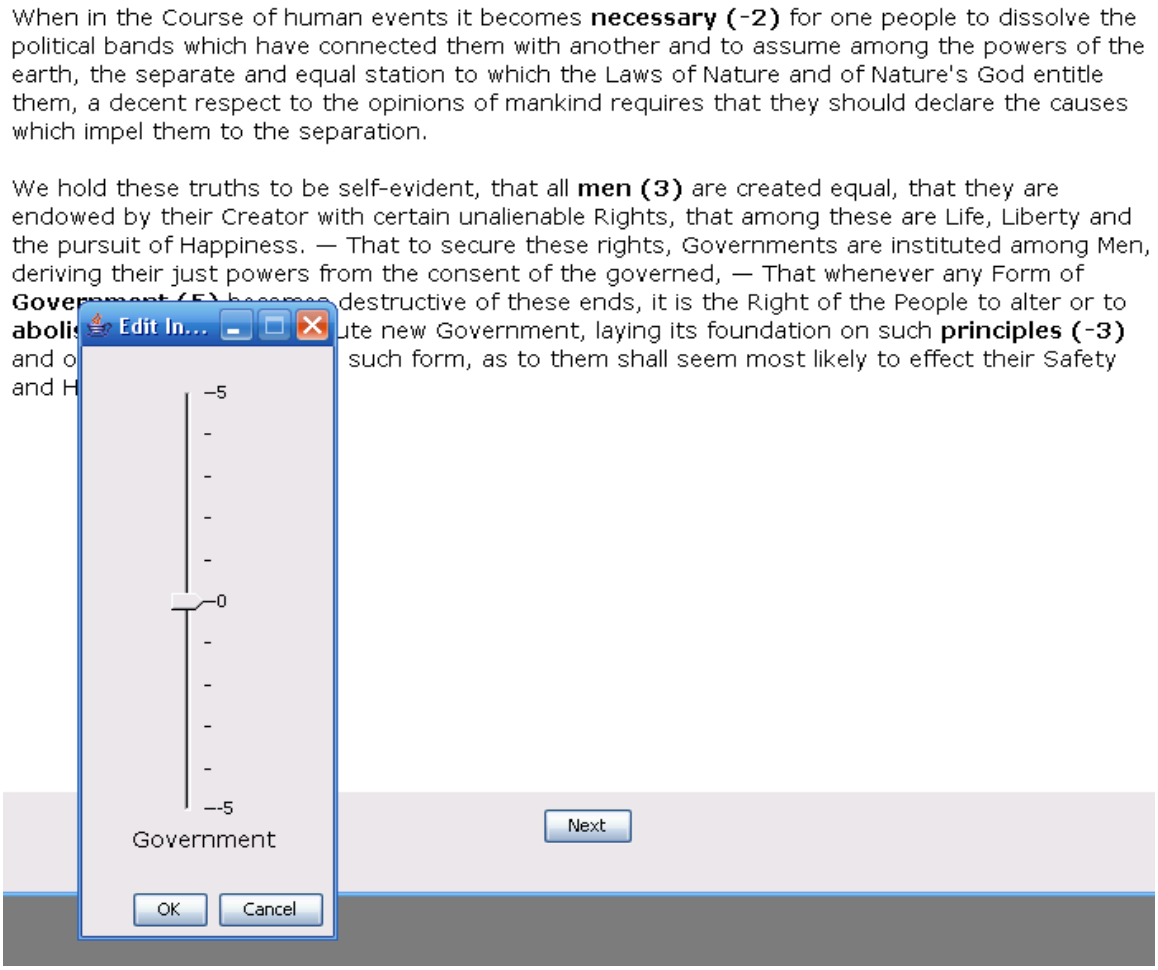


Figure 10: Rating words with the Typical Dialog Box Slider interface.

An instructional video explained how each interface works prior to interacting with the interfaces. Before beginning the tasks with each interface, the participants were given a short amount of time to practice using that interface. The experiment was a 2x2 within-subjects design where the independent variable is the interface used for the task. Order was also varied. All participants completed Task 1 first and Task 2 second. The interface conditions were counterbalanced, so that an equal number of participants used each interface first or second on each task. The mouse interactions of participants for

both interfaces in both tasks were logged. This enabled us to compute statistics about the times and answers for each condition.

Results - Quantitative

We measured how long it took participants to do each task with each interface. Of the 43 participants, 41 (95%) [X^2 (1) = 35.372, $p < 0.0001$] for Task 1 and 38 (85%) [X^2 (1) = 25.326, $p < 0.0001$] for Task 2 were faster at rating with the In-Context Slider (see Figure 11). Average completion time for Task 1 with the In-Context Slider was 72.39 seconds, while that of the Typical Dialog Box Slider was 122.68 seconds (see Figure 12). The difference was statistically significant [$F(1,42) = -13.263$, $p < 0.0001$]. Average completion times for Task 2 were 82.04 seconds with the In-Context Slider and 107.21 seconds with the Dialog Box Slider, and the difference between these is statistically significant [$F(1,42) = -4.535$, $p < 0.0001$] (see Figure 12). The accuracy measures for Task 2 for the two interfaces were not significantly different.

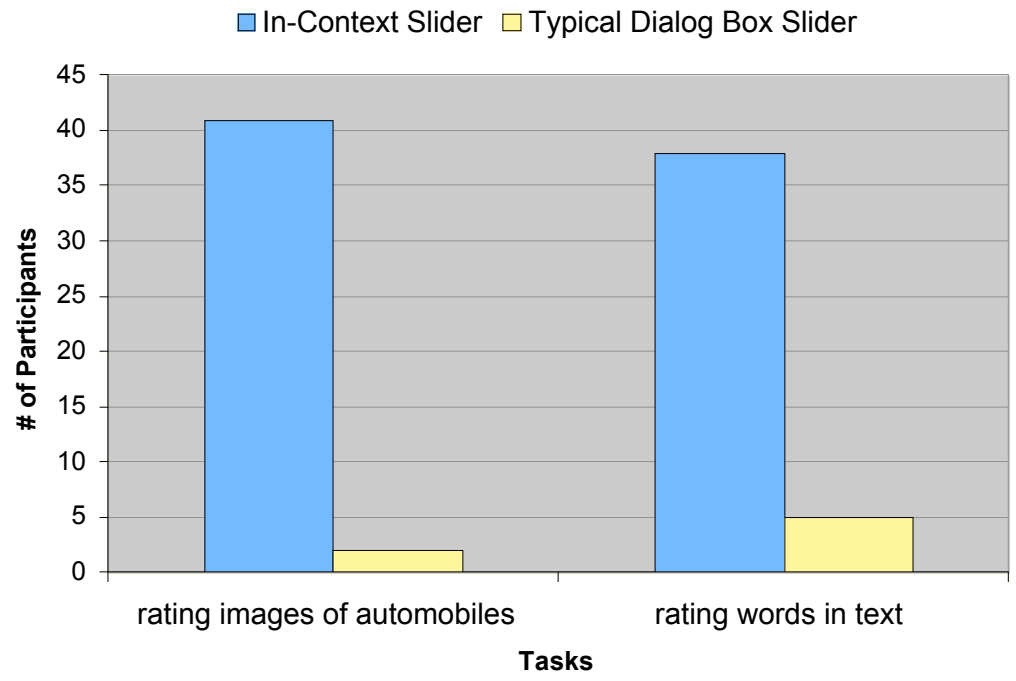


Figure 11: Time performance: which interface were participants faster with.

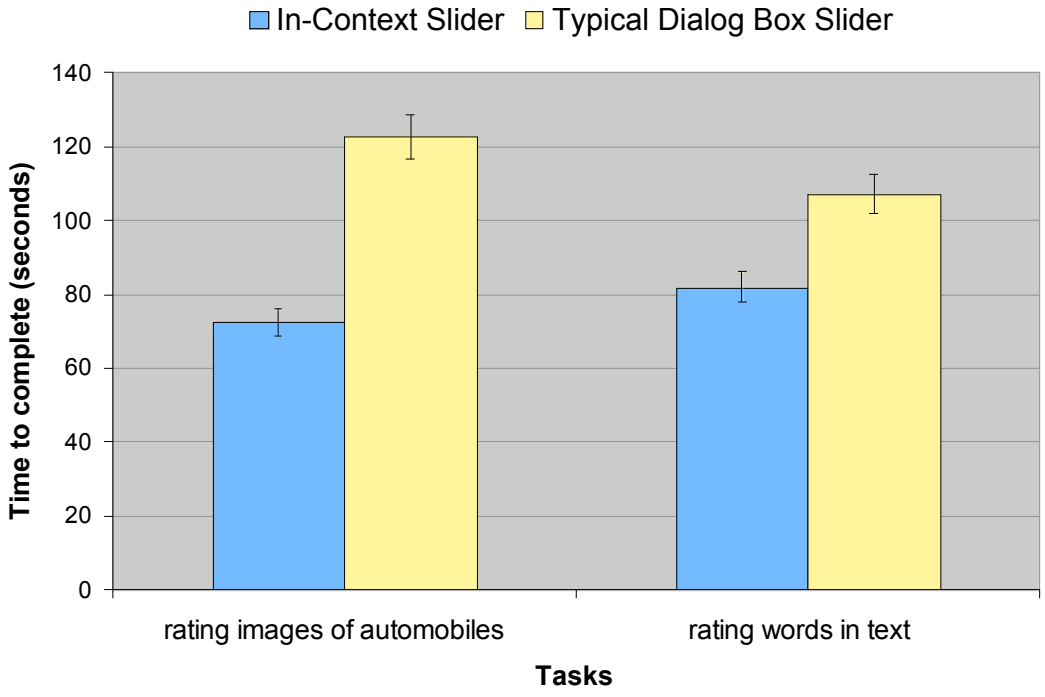


Figure 12: Time performance: average times to complete tasks.

We asked each participant which interface was easier to use. The possible responses were In-Context Slider, Dialog Box Slider, or both the same. For Task 1, 37 (86%) of the participants said the In-Context Slider was the easiest to use, and the results were statistically significant [$X^2(2) = 54.326$, $p < 0.0001$] (see Figure 13). For Task 2, 40 (90%) participants said the In-Context Slider was easiest to use [$X^2(1) = 28.488$, $p < 0.0001$]. Only one participant felt the Typical Dialog Box Slider was easier to use for Task 1.

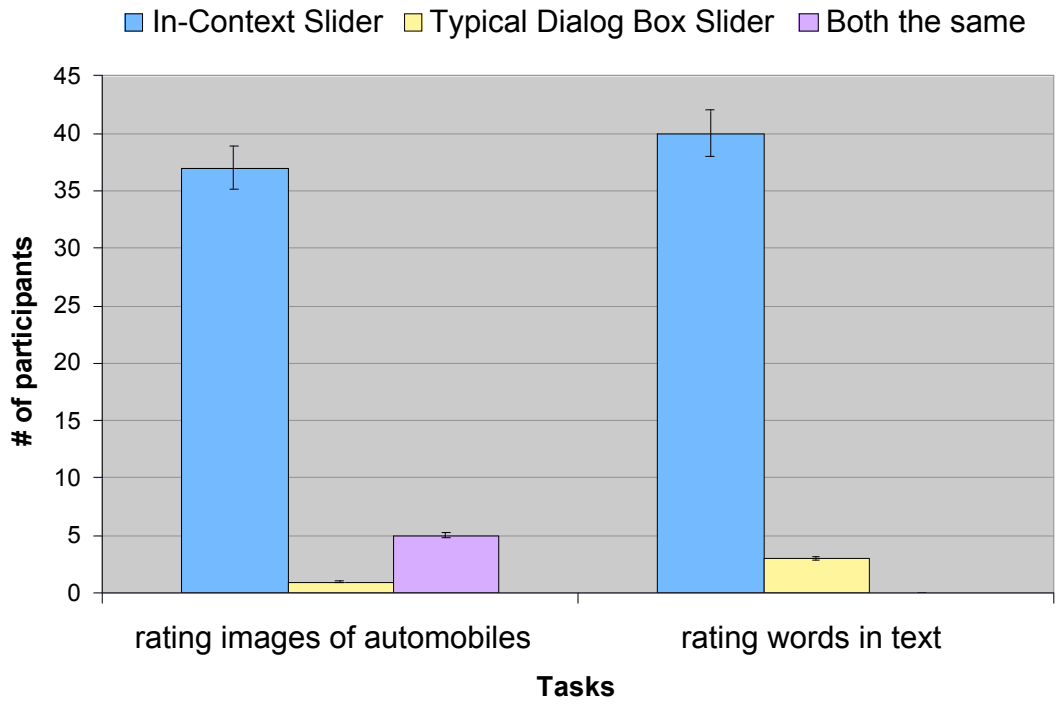


Figure 13: Participants' experience reports: which interface was easier to use?

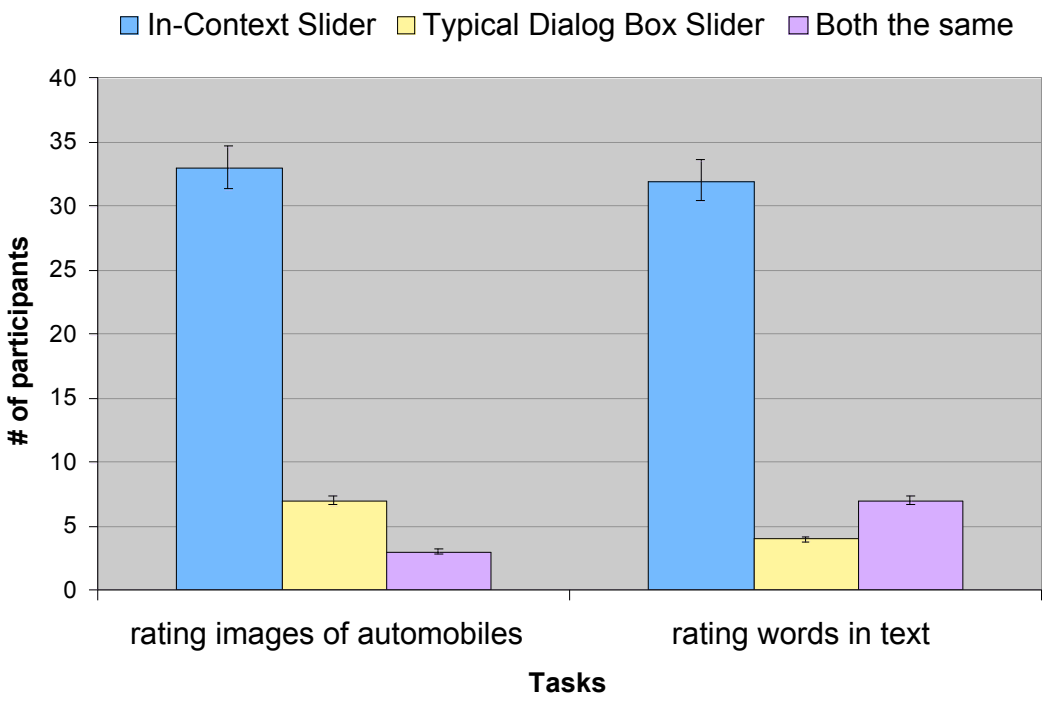


Figure 14: Participants' experience reports: which interface was more natural for expressing interest?

Participants were also asked which interface was more natural for expressing interest. Again, both the same was the third possible choice. From the 43 participants, 33 (76.7%) for Task 1 [X^2 (2) = 37.023, $p < 0.0001$] and 32 (74.4%) for Task 2 [X^2 (2) = 32.977, $p < 0.0001$] found the In-Context Slider to be a more natural interface for expressing interest (see Figure 14).

Results - Qualitative

The participants answered open-ended questions about their experiences, from which we obtained qualitative data. Many of the participants that found the In-Context Slider to be the easiest to use noted that the In-Context Slider required less effort to use in terms of mouse clicks.

> "The traditional slider was just more cumbersome to use. Having to right click then select your choice. The in context just seemed easier."

Several of participants recognized that the In-Context Slider's representation of values for interest level with red for negative and green for positive promoted comprehension.

> "It was just easier. The red and green helped identify the levels easier."

The colors also provided some participants with a realization of the affect of interest expression. To them, the experience of using the In-Context Slider was tied with emotional expressivity.

> "The colors made it easier to know how you felt. The pop-up was just setting a value while the in-context was almost setting an emotion."

Most of the participants that found the Typical Dialog Box Slider easier said that it was a more familiar interface for them. It was an interface that they were accustomed to or had used before; whereas, the In-Context Slider was a completely new and unfamiliar interface.

Discussion

The quantitative and qualitative results show that the In-Context Slider is quicker and easier to use than the Typical Dialog Box Slider. The In-Context Slider, through its fluid layers of activation, allowed the participants to more rapidly express interest with minimal distraction. The In-Context Slider's layer 0 and layer 1 activators provide less

disruption of the interactive space than the typical right-click popup menu. The sleek, precisely positioned, and translucent In-Context Slider layer 2 body is likewise designed to blend with and contribute to the participant's focus of attention within the interactive space, in contrast with bulky opaque dialog boxes that obscure context.

More than three fourths of the participants found the In-Context Slider to be a more natural interface for expressing interest than the Typical Dialog Box Slider interface. This result points out a problem with many of the standard interfaces for rating. These interfaces were designed primarily to obtain data for agent software, rather than to support human user experience. A human-centered design approach changes the experience.

The results are striking, considering that the In-Context Slider is a new interface, with which the participants had no prior experience. This was borne out by the qualitative data, in which most of the few participants who preferred the typical interface told us that they preferred it because it was familiar. This discrepancy, though not large, would be reduced in a realistic usage scenario longer than a 60 minute laboratory experiment. The ease of use findings are particularly significant since participants were not users with a particular background in interactive systems.

6 INTERGRATING INTEREST EXPRESSION WITH AUTHORING

In this section, we present the role of interest expression on the authoring process in combinFormation. We explain the problems with the previous interest expression interface in combinFormation, and introduce a new in-context interface that uses the In-Context Slider for interest expression. We describe the design and interaction of the different layer 0 activators in combinFormation.

Providing ratings of image and text surrogates, which visually represent documents and their constituent ideas, is an important part of the user interaction in combinFormation. combinFormation is a mixed-initiative creativity support tool that uses composition of images and text to represent collections of information resources [Kerne et al. 2006]. combinFormation uses two initiatives, the user and the agent (see Figure 15). The user directly manipulates the composition and the collection process through a set of design tools within the software. The agent semi-automatically collects and arranges within the composition image and text surrogates from online resources. A semantic model of the information resources and user's interests forms the basis for the agent's semi-automatic actions. The semantic model, which provides the basis for the effects of interest expression on the agents' actions, consists of two components: an information retrieval model and a hypermedia model. In character with the human-centered design of combinFormation, the user process of providing feedback that shapes the model is called "expressing interest," instead of "providing ratings." The user can express interest in an information object at any time, but never has to.

Prior versions of combinFormation provided a modal toolbar-based interface for interest expression. Among the problems with this interface was the need to look away

from the focus object, to the toolbar, in order to express interest. The goal of the design process was to create a better interface for interest expression in combinFormation while not disrupting the already existing authoring functionality within combinFormation. The In-Context Slider replaces the toolbar, creating a fluid interface that maximizes expressivity and minimizes cognitive load and task disruption through layered activation.

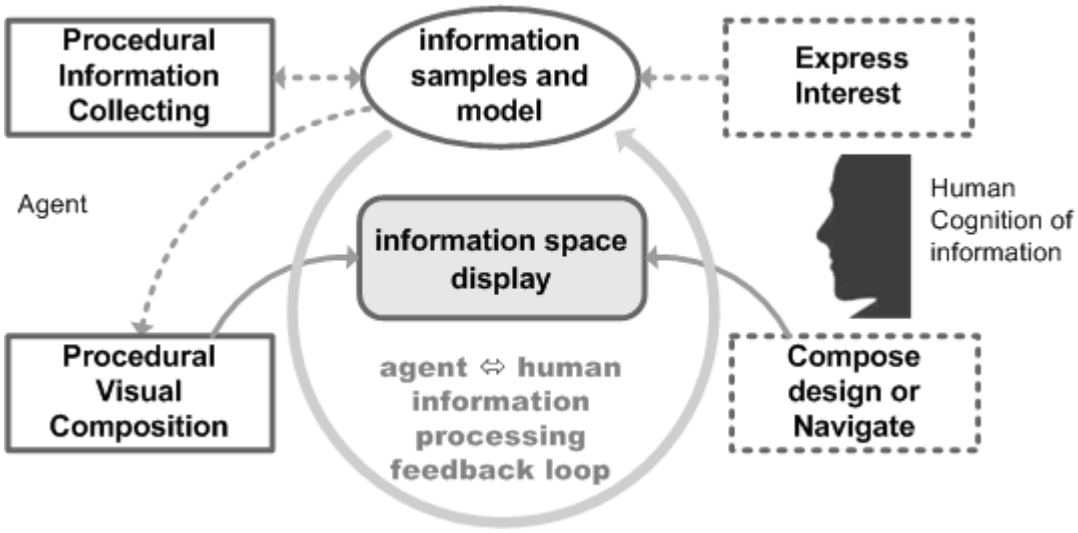


Figure 15: Feedback loop of agent and human information processing.

Authoring tasks with combinFormation involve conceptualizing, finding, editing, designing, and composing collections of information resources [Kerne et al. 2007]. The user is responsible for providing a set of *seeds*, normally at the beginning of a session, that is sent to the agent to start the collection process. A seed is an entity that provides the agent with the necessary details to locate information resources. Examples of seeds are search engine (e.g. Google, Yahoo) queries, RSS feeds, and website URLs. A user may also select from a curated collection of seeds. These curated collections include a news collection, a pop culture collection, and an art museum collection. The user's information

needs may evolve in the course of a session, in response to the spontaneous stimulus of found information. We call tasks in which the user's goal is to have ideas while collecting, *information discovery tasks* [Kerne and Smith 2004]. Information discovery tasks are divergent thinking tasks. Divergent thinking tasks are tasks that seek to answer open-ended questions where many different possible solutions exist. Information discovery is an iterative reformulation process where the flow of information is processed and mental models are formed and reformulated through browsing and composing relevant surrogates. These mental models can be run in mental simulations where unanticipated relevance or relationships are discovered.

combinFormation supports the user in information discovery tasks by using an agent to assist in the collection of information resources. However, the agent needs direction in order to effectively work in service to the user's information needs. Image and text clippings from documents in the composition space serve as affordances for interest expression, in addition to functioning as surrogates for the documents they come from.

Figure 16: combinFormation Double Modal Toolbar Interface. Left: Design Toolbar; Right: Interest Expression Toolbar.

Prior Double Modal Toolbar Interface for Interest Expression and Authoring

Prior to the In-Context Slider, interest expression in combinFormation was conducted on a per surrogate basis through a Double Modal Toolbar Interface. The Double Modal Toolbar affords setting two modes, a design mode and an interest expression mode (see Figure 16). Once set, these modes are combined when the user interacts with a surrogate in the composition space. For the design mode, the user selects among a set of tools for design interaction with a surrogate. These tools are a grab tool for repositioning surrogates in the composition space, a cut tool for removing surrogates from the composition space, a text tool for creating and editing text surrogates, and a navigate tool which when used on a surrogate opens in a web browser the document represented by the surrogate. The interest expression mode determines whether a positive, neutral, or negative interest is applied when using one of the tools in the design mode on a surrogate. The value of the interest expression mode is set by selecting positive, neutral, or negative on the Double Modal Toolbar. As a result of this double modal design, the user was constantly looking back and forth between surrogates and the toolbar.

The interest expression mode is eliminated by the In-Context Slider by providing each surrogate with its own In-Context Slider for direct user interest expression. The In-Context Slider, as an interest expression mechanism in combinFormation, was developed to reduce the burden on the user when providing feedback to the agent. The previous interface demanded more of the user in terms of time and attention. By reducing these demands, we hypothesized that the user would be more willing to provide feedback, thereby improving the agent's ability to obtain and show more relevant results. Removal

of interest expression mode from the toolbar resulted in the relocation of some of the modal tools to an in-context location around a surrogate. These changes along with the introduction of the In-Context Slider form the new interface for combinFormation, called the In-Context interface.

Some interactive affordances were provided around the surrogate in the prior interface (see Figure 17). These are details-on-demand, edit palette, latch, and search tool. Their presence represents an initial move toward providing an in-context interface, which the present research completes. Their functionality is described as part of the next section.

Figure 17: Positioning of details-on-demand, in-context tools, edit palette, and Surrogate In-Context Slider for Double Modal Toolbar interface.

Figure 18: In-Context Interface components (clockwise from top): details-on-demand, edit palette, tools, and Surrogate In-Context slider navel.

In-Context Interface for Interest Expression and Authoring

In the new In-Context Interface for combinFormation, when the user mouses over a surrogate, a set of interactive components are displayed around it. The surrogate is a clipping, which is an image or a sentence of text. As surrogates, the clippings function as visual representations of a document that link back to that original document. The interactive in-context components are the rollover frame, details-on-demand, direct manipulation tools, the edit palette, and the Surrogate In-Context Slider navel (see Figure 18). We review these components, and then provide details about how In-Context Sliders can be activated.

The rollover frame is a stroked outline with square boxes in the corners visualizing the boundary of the surrogate. These boxes serve as points for adjusting the size of a surrogate. The positioning of these visual interactive features depends on both the surrogate's position in the composition space and which interface is being used.

In-Context details-on-demand is an interactive set of text fields that contain metadata about the surrogate [Kerne et al. 2006]. As combinFormation's agents collect image and text surrogates, they also gathers metadata about each surrogate, such as the caption for an image, the title of the document, and additional semantic fields, when available, such as author and keywords. This metadata is displayed as fields in details-on-demand. These text fields can be edited by the user. All text fields are initially represented by a single line. Text fields having additional text contain "…" at the end of the line to visualize that additional text exists but is not being displayed. The additional text is shown by mousing over the text field which after a short delay expands the height of the text field to contain all the necessary lines to visualize the entire contents of the text field. The delay exists for the same reason that the appearance of a layer 1 activator for an In-Context Slider is delayed, to prevent expansion of the field when the user is simply passing through the field. Mousing out of an expanded text field causes the field to collapse back to a single line of text.

Each of the in-context tools directly manipulates the surrogate. The functionalities supplied vary between the In-Context Interface and the Double Modal Toolbar Interface. In both interfaces, there is a latch tool when toggled on prevents the agent from removing the surrogate and a synthesized Google search tool which when clicked provides a new Google search query seed to the agent. The In-Context Interface moves the cut tool and navigate tool from the design toolbar of the Double Modal Toolbar to the in-context tools, allowing in-context removal of surrogates and navigation to the document represented by the surrogate.

The edit palette provides functionality for adjusting the appearance of a surrogate. The available controls differ depending on whether the surrogate is an image or text. If the surrogate is an image, the edit palette toggles on and off translucency around the edges of the surrogate. This translucency allows for smoother edge transitions between the image surrogate and other surrogates. For a text surrogate, the edit palette allows changing the font face and size and the background color.

combinFormation Layer 0 In-Context Slider Activators

In combinFormation, there are four layer 0 activators. These layer 0 activators are the image and text surrogates, the words within a text surrogate, the words within a details-on-demand field, and an entire details-on-demand field.

Image and Text Surrogates as Layer 0 Activators

The first layer 0 activator in combinFormation is an image or text surrogate (see Figure 19a,b). The In-Context Slider adds a fourth object to position in-context of a surrogate. With each word within a text surrogate acting as a layer 0 activator, the layer 1 navel cannot be positioned on top of a surrogate.

We considered placing the Layer 0 activator in the middle of the surrogate, to minimize the user's effort in activation. However, placing the navel on top of the surrogate might also occlude important details of the surrogate, creating interaction problems, especially for text surrogates, because for text each word is also a Layer 0 activator. The In-Context Slider for a surrogate is always vertically centered on the left side of a surrogate (see Figure 19c,d). This of course means that surrogates positioned on the far left of the composition space may have inaccessible In-Context Sliders. This

positioning algorithm is a simplistic first iteration approach that will be replaced by a more complex, carefully designed approach later.

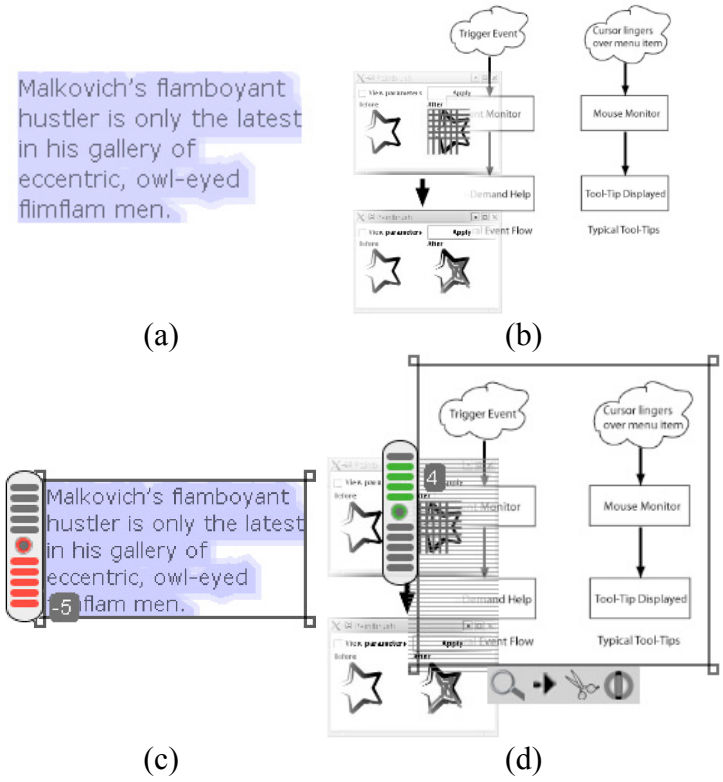


Figure 19: combinFormation surrogates (a) text surrogate, (b) two overlapping image surrogates, (c) fully activated text surrogate, (d) one of two overlapping image surrogates is fully activated.

Words as Layer 0 Activators

In combinFormation, both words inside a text surrogate and words inside fields in details-on-demand are layer 0 activators. The terms from within a text surrogate and from the metadata are used by the agent through the semantic model to determine what new surrogates to bring into the composition. The model stores interest values for each term that is not considered a stop word. Stop words are words that are too common to be

useful to the agent (e.g. the, an, is, are). A dictionary of stop words that we created is provided to combinFormation. As the interest values in terms change, the agent looks to obtain surrogates whose metadata (and self for text surrogates) contains terms with higher values. It discards those with lower values. Prior usability studies have indicated that the user needs the capability to directly affect the interest model on a per term basis in order to gain relevant and interesting results from the agent.

The second layer 0 activator in combinFormation is a word inside a text surrogate. The layer 1 navel for the words is the half-circle version. It appears below the word as long as the font size is not too small to provide space for a navel. Stop words are a special case because they cannot have interest expressed in them. In order to maintain consistency and make clear which terms are stop words, on mouse-over a crossed out navel is shown under a stop word (see Figure 20). When the crossed out navel is moused over, the layer 2 slider body does not appear because interest cannot be expressed in stop words.

When an In-Context Slider is activated, it immediately shows the current value associated with the Layer 0 activator. To better visualize interest expression values, when the Layer 0 activator is a word, the color of the activated word is also changed to match the color in the slider (see Figure 21). After a term is moused over and the In-Context Slider navel is shown, the term's color changes to match the current value of interest for that term. Words are often repeated or have multiple forms based on different stems, e.g. happy and happiness. Thus, within a text surrogate, the color for all instances of a word and its derivational forms whose value is being changed by an In-Context Slider also

change to match the value (see Figure 22). When the In-Context Slider is deactivated, all terms whose color changed revert back to their original color (i.e. black).

We have developed further fluid techniques to enhance legibility during in-context interactions. Text surrogates have colored backgrounds. These colored backgrounds can cause visibility issues when changing the color of the text. A lack of contrast in hue can make this text unreadable. To handle this issue, the background color of a text surrogate is desaturated (faded out) when changing the value of an In-Context Slider adding a contrast of saturation between the text color and the background color (see Figure 21). In the case of gray, which is already desaturated, the brightness is adjusted if necessary to guarantee that the text is always readable.

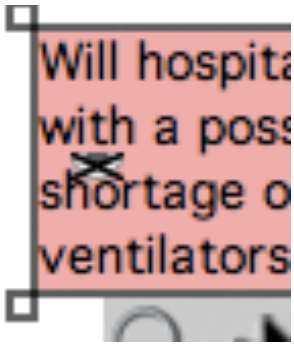


Figure 20: Crossed out navel of In-Context Slider for stop words.

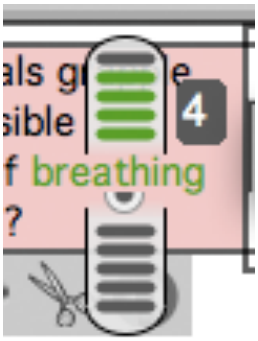


Figure 21: Text activated In-Context Slider changing the color of the text to reflect the current value.

Figure 22: Text surrogate with In-Context Slider adjusting interest in a word and changing the color of related terms within the text surrogate.

The third layer 0 activator is a word within a text field in details-on-demand. These words act identical to the words with text surrogates, except the white background prevents needing to change the background color when adjusting the value (see Figure 23).

Since both the words in details-on-demand and in text surrogates are In-Context Slider layer 0 activators, multi-activation is possible between each of these types of activators. The color changes in related words are also applied in both activated objects.

The semantics for changing the interest value for a metadata label are different from the others. Changing the interest in a label implies changing interest in that entire field. These semantics are visualized by changing the color of all affected non-stop words in the Interactive Metadata field that when a label's interest value is being adjusted with the In-Context Slider.

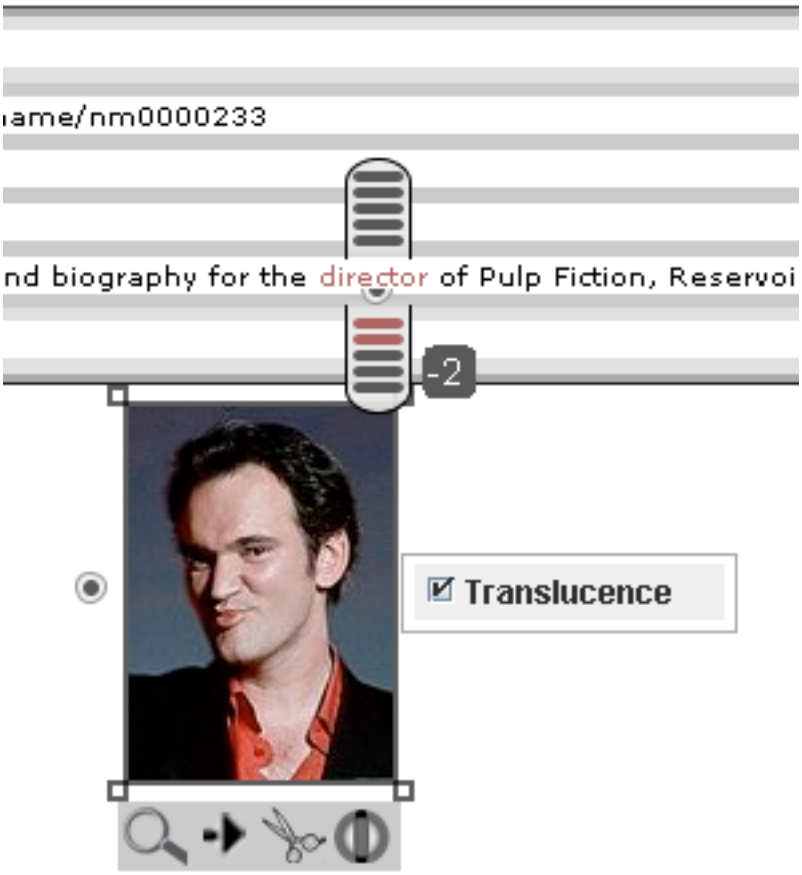


Figure 23: Example of activated In-Context Slider for a word in details-on-demand field.

Details-on-demand Fields as Layer 0 Activators

The fourth layer 0 activator in combinFormation is an entire field in details-on-demand. Each field in details-on-demand has a label. The label serves as the layer 0 activator. When a person moves the mouse cursor over the field label, a half-circle navel appears directly below it. Activating to layer 2 and adjusting the value causes each visualized word in the field that is not a stop word to change color to match the current value (see Figure 24). Currently in combinFormation, expressing interest in a details-on-demand field with the In-Context Slider applies that amount of interest to each non-stop word in a field. This methodology has a problem. There are different semantics involved in expressing interest in a field. Expressing interest in a field can mean more than just expressing an equal amount of interest in each of the individual words for that field. For example, one of the fields could represent the author's name of the research paper that the surrogate represents. Expressing positive interest in an author field means a desire to see more documents from a specific author. While changing the interest value for the words that form the author's name may have some of the desired affect, the agent cannot truly model the user's interest without knowing the semantics involved. The agent will return documents that contain that author's name including documents not created by the author. Different semantics are needed to address this issue, but are outside the scope of this research.

Figure 24: Example of activated In-Context Slider for details-on-demand field label.

7 EVALUATION – EXPERIMENT 2

The In-Context Slider is designed as expressive interface. It serves as an interest expression mechanism for combinFormation and has a direct impact on the experience of the user when completing information discovery tasks with combinFormation. This experiment was designed to evaluate the In-Context Interface for information discovery tasks. We hypothesized that the new In-Context Interface would better help promote information discovery and ideation in comparison with the previous Double Modal Toolbar Interface.

Participants

Twenty-two subjects participated in this experiment. Once again, the subjects were students from an introductory psychology course. This was a different set of subjects than those who participated in the experiment reported above.

Method

Participants were asked to complete two information discovery tasks using combinFormation. They used the In-Context interface for one task, and the Double Modal Toolbar interface for the other. The interfaces were counterbalanced across participants, so that half the participants used the In-Context interface first while the other half used the traditional interface first. The two information discovery tasks were:

- Your department adviser has suggested participating in a summer internship. What would you enjoy doing for a summer job? Where would you work?

- If you could spend a semester studying anywhere in the world, where would you choose to go? What would you study while there?

These two tasks were carefully selected because of their similar levels of personal interest for the participants, undergraduate college students. Below each of these tasks descriptions were the following two paragraphs elaborating how the participants were to answer these questions:

> Use the composition space to develop and explain your answers. Collect information about options that you are considering. Brainstorm. Refine ideas. Reference appropriate information. Develop a composition that you could show to others to explain your ideas.
>
> Use which ever interest expression mechanism is provided (either the In-Context Slider or the interest expression tool in the toolbar) to get more relevant results. Express positive interest in useful things, and negative interest in unhelpful things.

Prior to doing each information discovery task, participants were shown an instructional video explaining how to use combinFormation with a given interface. The video for the second task contained only an explanation of the changes between the two interfaces. The participants were given a brief warm-up session to gain familiarity with combinFormation and the interface. The participants were given 22 minutes to complete each task. The final compositions were logged for each participant on both tasks. The

total numbers of changes to interest expression for each task were also logged. After completing each task, the participants were asked a series of questions about their experiences during that task. Participants were asked to rate on a Likert scale how interesting and relevant were the surrogates brought by the agent, as well as, how affective was their interest expression on the agent. Upon completing both tasks, participants were asked comparison questions about which interface was easiest to use and which was easiest to express interest with.

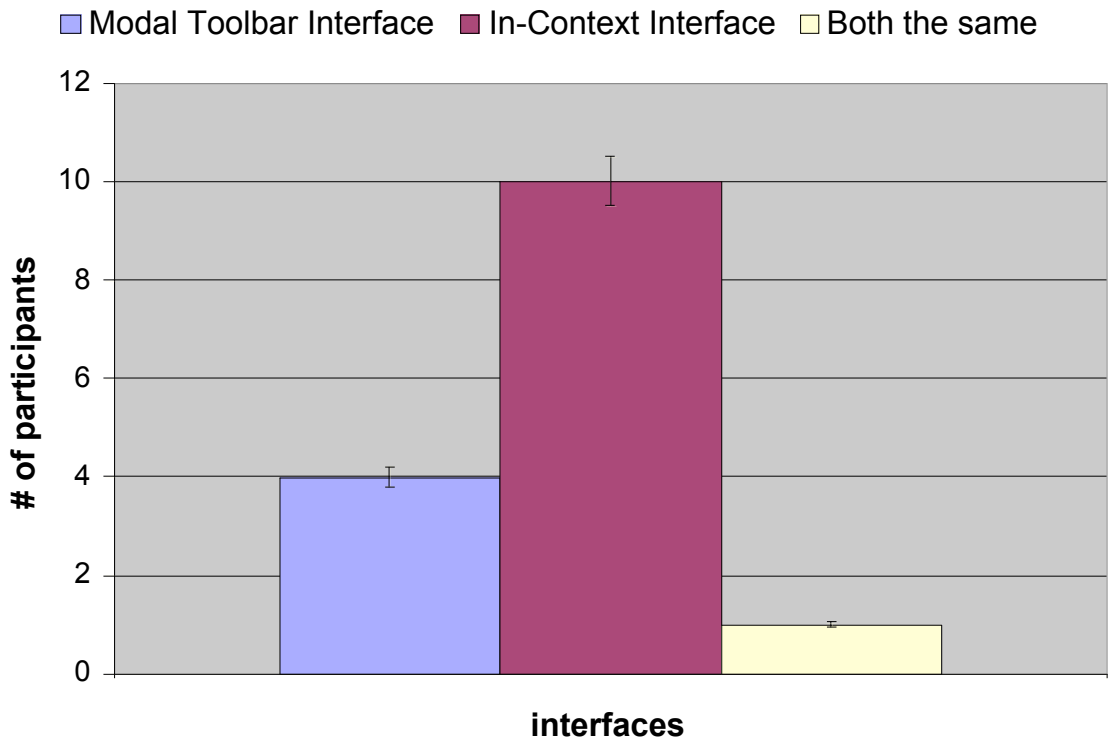


Figure 25: For participants who expressed interest: which interface was used more for interest expression? Also, which interface was easier to use for expressing interest?

Results – Quantitative

Of the twenty-two participants, ten (45%) of the participants used the In-Context interface more for interest expression. Another eight (36%) expressed interest an equal number of times with the two interfaces where seven of the eight expressed no interest at all with either interface, and finally, four (18%) expressed more interest with the Modal Toolbar interface. Taken altogether, these results indicate a trend, but were not statistically significant. However, when we exclude the participants who didn't use either interface, and consider the 15 who did express interest, ten (67%) expressed interest more with the In-Context interface, 1 (6%) expressed the same amount with both, and 4 (27%) expressed interest more with the Double Modal Toolbar (see Figure 25); the result is statistically significant [$X^2(2) = 8.4$, $p < 0.015$]. Further, among these participants, the average number of changes to interest expression was 8.1538 for the In-Context interface and 5.3846 for the Double Modal Toolbar interface (see Figure 26). This result is also significant [$F(1,12) = -2.241$, $p < 0.045$].

These objective results correlated directly with the participants' reports on their own experiences. Ten subjects said the In-Context interface was easiest to express interest with, one subject among those who expressed interest said the interfaces were equally easy to express with, and four subjects said the Double Modal Toolbar was the easiest [$X^2(2) = 8.4$, $p < 0.015$].

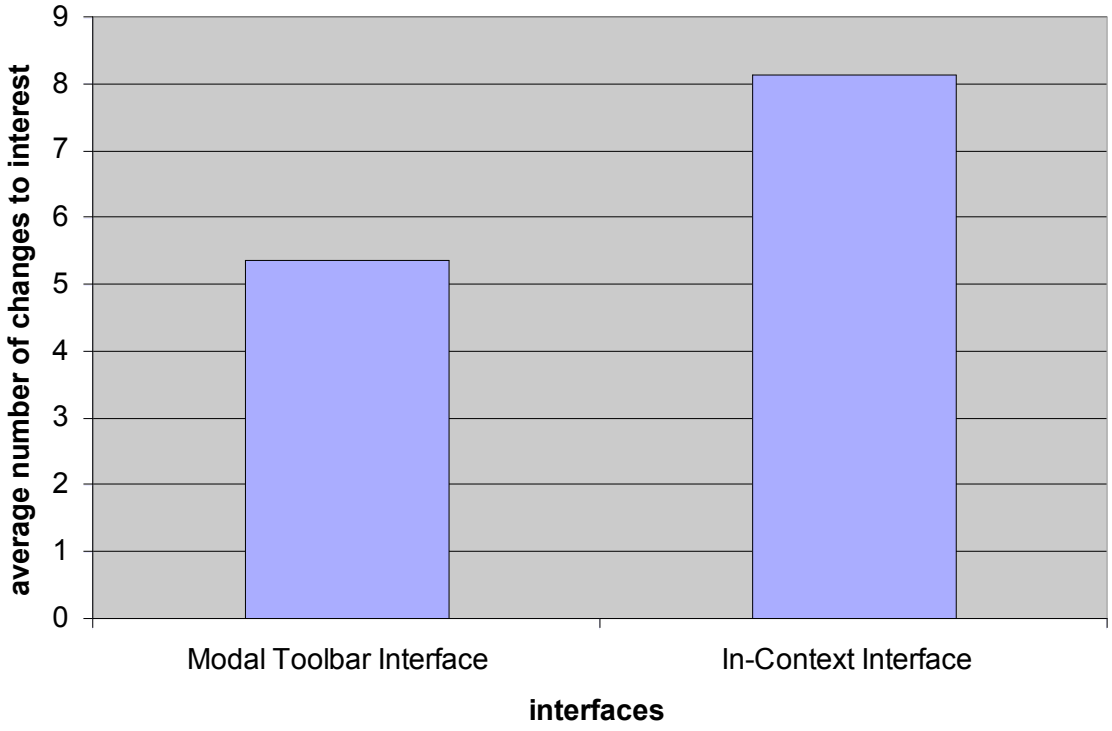


Figure 26: Expressions of interest. For participants who expressed interest, how many times did they change the interest level with each interface?

The information discovery tasks were evaluated for creative ideation using the information discovery measures developed in [Kerne and Smith 2004] and [Kerne et al. 2007]. In the Information Discovery Framework, objective measures are applied to the answers of information discovery tasks. The objective measures are emergence, flexibility, quality, and originality. Emergence represents the formulation of a new idea from the combination of other ideas. Flexibility is a measure of the variety of a set of answers. Quality measures how interesting or valuable an answer is. Originality looks at how unique an answer is. This measure is derived by developing a master list of all answers, and then using this list like inverse document frequency [Baeza-Yates and Ribeirno-Neto 1999] to determine uniqueness. We count the number of participants that

provide a specific answer. Answers that are given by fewer participants are given higher originality scores. Flexibility is measured by looking at the total number of websites represented by surrogates in a composition.

The criteria for scoring emergence and quality are shown in Table 1. These criteria were designed to be mutually independent. Quality measures how well the participant answered the question. Emergence measures how groups of collected surrogates and annotations contribute new ideas that are not found in the original surrogates, themselves.

Table 1: Emergence and quality metrics used for rating participants' answers to information discovery tasks.

	Score	Criteria
Emergence	0	The subject assembled elements to answer a given question, but recognizable relationships and new ideas are minimal.
	1	Coherence between elements but not otherwise unrelated (uniform theme) –or- new relationships between elements but no coherence.
	2	Otherwise unrelated elements in a coherent group.
	3	Otherwise unrelated elements in a coherent group(s) in a way that is clear and insightful.
Quality	0	Answer seems to have no relation to the question
	1	Some relevance. Little or no explanation.
	2	Multiple perspectives through elements.- Some explanation.
	3	Brilliant – Wow, that was very interesting. Better explanation.

The quantitative values derived from applying the measures were compared between all participants' answers (compositions) to the two information discovery tasks. Emergence, flexibility, quality and originality scored comparable results with both interfaces. The differences in each of these ratings between interfaces are not statistically significant.

We have seen significance in the results for interest expression frequency and user experience reports, for users who expressed interest as compared to those who did not. We recorded further significant results with this basis in the information discovery measures of quality and emergence. For quality, the average rating for participants who expressed interest was higher than (mean 1.7308) for those who expressed interest, as compared to those who did not express interest at all (1.000). These differences are statistically significant [$F(1,42) = 3.533$, $p < 0.001$]. Interest expression leads to higher quality composition, according to the criteria in Table 1.

For emergence, among those who expressed interest, using the In-Context interface resulted in better performance, that is, in new ideas. The average emergence rating for the In-Context interface when the participants expressed interest was 1.5833; whereas, the average emergence rating for the In-Context interface for the participants that did not express interest was 0.8000 [$F(1,20) = 2.313$, $p < 0.032$]. The average emergence rating for the Double Modal Toolbar interface was 1.2857 when participants expressed interest and 0.8750 when participants did not express interest.

For the series of questions following each task where the participants were asked to rate on a Likert scale how interesting and relevant were the surrogates returned by the agent, and to what effect did interest expression have on the agent, the results showed no

statistically significant difference between the two interfaces. Both interfaces scored comparably on each question.

Results - Qualitative

We collected qualitative data regarding the participants' experiences. Figure 27 depicts an example composition created by one participant for the internship information discovery task. The composition shows the participant is interested in an internship at a dentist's office. In particular, the participant seems interested in children's dentistry. Many of the images depict children in a dentist chair or displaying bright smiles. Several of the textual elements point to information about dentistry jobs. Several other examples of compositions created by participants can be found in Appendix A.

We collected comments about the experience through open-ended questions. Several participants reported that the In-Context interface was easier to use since interest expression did not involve continually moving back and forth between the toolbar and the object of focus.

> "[The In-Context interface] was easy to express interest with because you could do it on the fly without having to go back and choose your interest each time."

> "I could easily rate the picture I selected because the [navel] would immediately open instead of a tool bar where I had to click elsewhere and a few more times."

Figure 27: Composition of surrogates created by a study participant for the study abroad information discovery question. An In-Context Slider can be activated for each surrogate and each word.

Discussion

The results from Experiment 2 show that the In-Context interface is an easier interface for expressing interest within combinFormation, in comparison to the Double Modal Toolbar interface. This was reflected consistently, among those who expressed interest, in the quantitative results for which interface was used more for expression, in the number of times they expressed interest, and in their experience reports. The fact that some participants did not express interest at all is not surprising. The form of the study is a brief exposure to a new interface paradigm, the combinFormation composition space. Thus, it is reasonable that some participants did not completely understand the need for expressing interest to affect the agent. The motivation level of the anonymous

undergraduate subjects, none of whom we knew personally, could also be an issues, since the agent will automatically provide some result (although not necessarily a good result).

The lack of difference between the two interfaces in terms of relevance and interestingness of what the agent was able to collect as reported by participants' experiences points to possible problems in how interest expression affects the agent. In the qualitative comments, some participants felt that interest expression had no effect on the agent, and that the seeds provided at start-up had more of an affect than anything else on what the agent returned. This indicates that the structure, function, and implementation of the interest model and how the In-Context Slider affects that model require closer examination. This issue more than likely resulted in some participants expressing less interest using both interfaces, because they felt disenfranchised from the agent.

> "I think that the major difference between the two interfaces was the search query [seeds]. When I searched for jobs a lot of junk came up which made it difficult to find what I was looking for. When I searched for information on a location, it was much easier to find relevant pictures."

However, the results did convey benefit from interest expression on information discovery tasks. Comparing the rating measures of participants who expressed interest versus those who did not reveals that those who expressed interested had answers with higher quality for both interfaces and higher emergence for the In-Context interface. Since participants who expressed interest showed greater emergence and quality in their solutions, the role of interest expression in combinFormation's function as an information

discovery tool is an important one. Through the quality and emergence measures, interest expression promotes creativity. Therefore, the interest expression mechanism for combinFormation should encourage interest expression. The results show that the In-Context Slider is an easier to use and more frequently used interface compared with the interest mode of the Double Modal Toolbar interface, making the In-Context Slider a better interface for interest expression in combinFormation's mixed-initiative composition space.

Achieving significant results for information discovery measures across conditions in a complex mixed-initiative system like combinFormation is difficult. It is difficult to design study protocols that isolate significant independent variables. There are many factors in the program's operation, such as the semantic model and agent structures, and the interactive interface components for authoring and directing the agent. Further, study participants access the entire Internet to form answers to questions. The quality of available information from web sites and search engines, and the download times, are highly variable.

Nonetheless, the result for emergence using the In-Context Slider was significant. The fluid In-Context Slider interface enhanced the creativity of the participants. We generalize this important finding to conclude that fluid interface components for expression can enhance creativity. Further development of such interfaces deserves further research.

8 CONCLUSION

Many of the current interest expression interfaces require extra effort and attention on the part of the user. These interfaces are often activated through a series of menus or keyboard commands and located in a popup window or a side bar that is not always located near the object of interest. Or, they use dedicated web-based forms with slow responses. Thus, the user is reluctant to use the interest expression interface. Fluid In-Context interfaces are appropriately suited for interest expression mechanisms. The minimal effort required to use these interfaces can reduce the unwillingness of users to express interest. A user's decision about the relevance of information occurs while that information is in the user's focus. Having an interest expression mechanism appear in-context allows the user to express interest immediately and directly. Integration with authoring enables the user to focus attention on more primary tasks, and perform interest expression spontaneously when it feels worthwhile.

When Shneiderman and Bederson suggested increasing automaticity to help maintain user attention, they were referring to designing command sequences such as keyboard shortcuts that reduce the interactive steps required to complete tasks. With the In-Context Slider, as a fluid In-Context interface, we instead seek to increase automaticity through visual design. By designing simple, distinguishable visual affordances such as the navel, the user is able to quickly recognize interaction possibilities.

The navel is a small, simple and clear affordance providing visual continuity between un-activated and activated states. With the navel located in the center of an In-Context Slider, it places the mouse cursor at the center of interaction. The navel functions

as a focal point for interacting with an In-Context Slider. It helps the user learn what the slider does and how it works, forming a recognizable affordance, that when seen again, a user will understand its function.

User engagement in laboratory information discovery tasks using combinFormation with the In-Context Slider proved meaningful for personal growth and development. After viewing the compositions that participants created, it became clear that some participants, such as the creator of Figure 27, went through a thought provoking process in which they obtained information and synthesized ideas that may actually affect future decisions in their lives.

Authoring is an iterative process of creating, collecting, refining, and composing ideas. The process involves emphasizing certain ideas and discarding others. Expression is an important part of this process. When authoring with systems like combinFormation that use agents, expressing interest in relevant information is beneficial. Yet, it can take attention away from other task components. Thus, an interface for interest expression needs to minimize the demand on a user's attention allowing interest expression to occur easily as if expressed through the body and not through a disconnected interface. The full set of design choices for the slider: color, fluidity, translucence, integration, fluid gesture, and lack of saccadic movements produce an embodied sense of affect that promotes expression.

REFERENCES

BADDELEY, A.D. Is working memory working?, *Quarterly Journal of Exp Psych*, 44A, 1-31, 1992.

BAECKER R., SMALL, I., AND MANDER, R. Bringing icons to life. In Proceedings of *ACM CHI 1991* (New Orleans, LA), 1–6, 1991.

BAEZA-YATES, R. AND RIBEIRO-NETO, B. Modern Information Retrieval, Addison Wesley, New York, 1999.

BALABANOVIĆ, M. AND SHOHAM, Y. Fab: Content-Based, Collaborative Recommendation, *Communications of the ACM*, volume 40, number 3, 66-72, March 1997.

BEDERSON, B.B., HOLLAN, J.D., DRUIN, A., STEWARD. J., ROGERS, D., AND PROFT, D. Local tools: An alternative to tool palettes. In Proceedings of *ACM UIST* (Seattle, WA) *1996*, 169–170, 1996.

BIER. E.A., STONE, M.C., FISHKIN. K., BUXTON, W., AND BAUDEL, T. A taxonomy of see-through tools. In Proceedings of *ACM CHI 1994* (Boston, MA), 358–365, 1994.

EICK, S. Data Visualization Sliders. In Proceedings of *ACM UIST 1994* (Marina del Ray, CA), 119-120, 1994.

FITTS, P.M. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, volume 47, number 6, June 1954, 381-391.

GNU Image Manipulation Program. http://www.gimp.org (April 25, 2007).

GUIMBRETIERE, F. AND WINOGRAD, T. Flowmenu: Combining command, text, and data entry. In Proceedings of *ACM UIST 2000* (San Diego, CA), 213–217, 2000.

HA, V. AND HADDAWY, P. Toward Case-Based Preference Elicitation: Similarity Measures on Preference Structures. In Proceedings of *Conference on Uncertainty in Artificial Intelligence* (Madison, WI), 1998.

HENDERSON, D.A. AND CARD, S.K. Rooms: the use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface. *ACM Transactions on Graphics*, volume 5, number 3, 221-243, 1986.

Java. Sun Microsystems. http://java.sun.com (April 25, 2007).

KERNE, A. AND SMITH, S.M. The Information Discovery Framework. In Proceedings of *Designing Interactive Systems 2004* (Cambridge, MA), 357-360, 2004.

KERNE, A., KOH, E., DWORACZYK, B., MISTROT, J.M., SMITH, S.M., GRAEBER, R., CARUSO, D., CHOI, H., WEBB, A., AND JOSHI, P. A Mixed-Initiative System for Representing Collections as Compositions of Image and Text Surrogates. In Proceedings of *Joint ACM/IEEE Conf. on Digital Libraries (JCDL)* (Chapel Hill, NC), June 2006.

KERNE, A., KOH, E., SMITH, S.M., CHOI, H., GRAEBER, R., AND WEBB, A. Promoting Emergence in Information Discovery by Representing Collections with Composition. Proceedings of *Creativity and Cognition 2007* (Washington D.C.), in press.

KOIKE, Y., SUGIURA, A., AND KOSEKI, Y. Timeslider: An interface to specify time point. In Proceedings of *ACM UIST 1997* (Banff, Canada), 43–44, 1997.

KURTENBACK, G. AND BUXTON, W. The limits of expert performance using hierarchic marking menus. In Proceedings of *ACM CHI 1993* (Amsterdam, The Netherlands), 7, 1993.

MACKENZIE, S. AND BUXTON, W. Extending Fitts' Law to Two-Dimensional Tasks. In *Proceedings of ACM CHI 1992* (Monterey, CA), 219-226.

MCGUFFIN, M. AND BALAKRISHNAN, R. Acquisition of expanding targets. In *Proceedings of ACM CHI 2002* (Minneapolis, MN), 57–65, 2002.

MCGUFFIN, M., BURTNYK, N., AND KURTENBACH, G. FaST sliders: integrating marking menus and the adjustment of continuous values. *Graphics Interface*, 2002.

MCNEE, S.M., LAM, S.K., KONSTAN, J.A., AND RIEDL, J. Interfaces for Eliciting New User Preferences in Recommender Systems. In *Proceedings of the 9th International Conference on User Modeling* (Johnstown, PA), 178-188, June 2003.

NAGY, A.L. AND SANCHEZ, R.R. Critical color differences determined with a visual search task, *Journal of the Optical Society of America*, A 7, 1209–1217, 1990.

NORMAN, D. The Design of Everyday Things, Basic Books, New York, 1988.

SELLEN, A.J., LOUIE, G., HARRIS, J.E., AND WILKINS, A.J. What brings intentions to mind? An in situ study of prospective memory. *Memory*, volume 5, number 4, July 1997, 483-507.

SHNEIDERMAN, B. AND BEDERSON, B.B. Maintaining concentration to achieve task completion. In Proceedings of *ACM DUX 2005* (San Francisco, CA), 9, 2005.

TERRY, M. AND MYNATT, E.D. Side views: Persistent, on-demand previews for open-ended tasks. In Proceedings of *ACM UIST 2002* (Paris, France), 71–81, 2002.

TSANDILAS, T. AND SCHRAEFEL, M.C. User-controlled link adaptation. In Proceedings of *ACM HT 2003* (Nottingham, U.K.)*,* 152–161, 2003.

WARE, C. *Information Visualization: Perception for Design*. 2nd Edition. Morgan Kaufmann, San Francisco, CA. 2004.

ZELLWEGER, P.T., CHANG, B.W., AND MACKINLAY, J.D. Fluid links for informed and incremental link transitions. In Proceedings of *ACM HT 1998* (Pittsburgh, PA), 50–57, 1998.

APPENDIX A

The In-Context Slider is implemented in the Interface Ecology Lab Interactive Framework, an open-source Java library for building interactive applications and interfaces. The framework contains a collection of Graphical User Interface (GUI) objects that are reusable and extensible. Within this library is a Java package, named ecologylab.gui.incontextslider, containing all the necessary code for implementing the In-Context Slider in an application.

This package contains an abstract class called InContextSlider which is extended by two other classes in this package, ElementActivatedInContextSlider and TextActivatedInContextSlider. These two classes represent the two types of In-Context Slider layer 2 slider bodies implemented. Either of the two can be used or extended upon to handle unique functionality for specific applications. As well, new classes can extend InContextSlider to create new types of layer 2 objects. The In-Context Slider layer 1 navel is represented by the Navel class. This class implements both the full-circle and half-circle navels. Custom navels can be created simply by extending this class. The package also contains two Java Interfaces, InContextSliderActivator and TextInContextSliderActivator, which any visual GUI object implements to function as layer 0 activator. TextInContextSliderActivator extends InContextSliderActivator and provides additional structuring for textual GUI objects that need to activate an In-Context Slider. A class called InContextSliderSelection creates a GUI object for the multi-activation of In-Context Sliders. This class contains a data structure responsible for creating, storing, and removing In-Context Sliders in the multi-activation process.

For an application developer to add the In-Context Slider to an application, the steps necessary vary depending on whether the application can use one of the existing versions of the In-Context Slider or if a new version of the In-Context Slider needs to be created.

The first step for any case is to have all layer 0 activators in an application implement one of the activator interfaces. For single words or single lines of text, the TextInContextSliderActivator provides adequate functionality. A new version of the In-Context Slider may require an additional activator interface. This new activator interface should extend the InContextSliderActivator.

If the application requires a new version of the In-Context Slider, the application developer next creates a new class for this new version that extends InContextSlider and implements the abstract methods from InContextSlider. The implementation of the abstract methods should reflect the functionality and appearance of this new version of the In-Context Slider. The application developer may also need to override methods in InContextSlider, such as to greater adjust the functionality and appearance.

Once all required versions of the In-Context Slider are created, the application developer must next decide if multi-activation is of use in the application. If multi-activation is not needed, the developer can simply create individual instances of each In-Context Slider required. However, if multi-activation is needed, the developer either instantiates InContextSliderSelection if custom functionality is not required or creates a new class that extends InContextSliderSelection. The application developer then overrides each method to acquire the desired functionality.

InContextSliderSelection contains an inner class called InContextSliderPool that keeps track of all In-Context Sliders in a selection. IncontextSliderPool has a method called nextAvailable(), which is passed an argument of enum type InContextSliderVersion. This method returns an InContextSlider instance of the appropriate type. nextAvailable() is called by InContextSliderSelection in the method newSliderAtActivator(). The InContextSliderVersion that is passed is provided by the activator through a method declared in the InContextSliderActivator interface called inContextSliderVersion(). If a new version of the In-Context Slider that needs multi-activation is created, an enumeration needs to be added to InContextSliderVersion, and a case statement added to the switch in nextAvailable() that instantiates a version of the new In-Context Slider.

The In-Context Slider is connected to a value using the ecologylab.gui.Scaled-ValueObserver interface. The In-Context Slider uses the model-view paradigm through the Observer and Observable classes in Java. ScaledValueObserver extends Observer with a single method, getScaledValue(). This method returns a scaled value that is used by the In-Context Slider to get the current value to visualize. A class that contains a value that needs to be adjusted by an In-Context Slider should implement ScaledValueObserver. Then, this class needs to be added as an observer to the appropriate InContextSlider object by calling within InContextSlider either the method addObserver() for appending to the observer list or the methods changeObserver(ScaledValueObserver) for replacing all observers with the single observer passed in and changeObserver(ArrayList<ScaledValueObserver>) for replacing all observers with a set of observers.

When a value is set using the In-Context Slider, the InContextSlider object calls notifyObservers(Object) and passes in the new value. Each observer then updates its value using its corresponding update(Object, Object) method. This functionality allows an In-Context Slider to affect multiple objects and allow each of those objects to handle that effect its own way.

Example Implementation: In-Context Slider within combinFormation

combinFormation has four layer 0 activators. The activations of these In-Context Sliders are handled by two classes in combinFormation called cf.gui.SurrogateInContext-Slider and cf.gui.CFTextInContextSliderSelection. SurrogateInContextSlider represents the In-Context Slider for a surrogate. CFTextInContextSliderSelection represents all other In-Context Sliders within combinFormation. A number of different classes will be described in this section. Table 2 shows these classes and the corresponding classes in the Ecology Lab Framework that are either extended or implemented by the combinFormation classes.

Table 2: Ecologylab package classes and their corresponding implementations or subclasses in combinFomation

ecologylab.gui.incontextslider classes	*combinFormation classes*
ElementActivatedInContextSlider	SurrogateInContextSlider
InContextSliderSelection	CFTextInContextSliderSelection
InContextSliderActivator	Surrogate
TextInContextSliderActivator	TextTokenGUIt IMTextTokenGUIt TextChunkTokenVisual

combinFormation requires one new version of the In-Context Slider, SurrogateIn-ContextSlider. This version shares no visual differences with ElementActivatedIn-

ContextSlider, but does contain several functional differences related to activation and interaction. SurrogateInContextSlider extends ElementActivatedInContextSlider and adds functionality for expressing interest in a surrogate. After expressing interest using a SurrogateInContextSlider by setting the value with a left mouse button click, the mouse cursor is located outside the surrogate. Special interactive functionality is provided to prevent the removal the surrogate's in-context interactive objects (e.g. details-on-demand) including the SurrogateInContextSlider. The slider simply reverts to the layer 1 collapsed form. If the mouse cursor is moved away from the surrogate outside the bounds of the expanded form of the In-Context Slider, all in-context objects are removed, while if the mouse cursor is moved toward the surrogate the in-context objects remain.

CFTextInContextSliderSelection extends InContextSliderSelection. All In-Context Sliders in combinFormation that are activated by a word are represented by this class. Included in this class are methods for changing the color of related words. A HashMap is used to store references to the related GUIt objects in both details-on-demand and a text surrogate. The HashMap uses the stems as keys and stores an ArrayList of GUIts for each key. The HashMap allows for quick access to the necessary GUIt objects when changing the color of text. This HashMap is reconstructed with each reconstruction of details-on-demand that occurs whenever a new surrogate is moused over for a set amount of time.

Each word in both details-on-demand and a text surrogate extend the class cf.gui.TextTokenGUIt. TextTokenGUIt implements TextInContextSliderActivator and contains generic interactive functionality for activating an In-Context Slider for a word

within combinFormation and visualizing the value of that slider. The activate method in TextTokenGUIt contains the following code:

```
if (shouldActivate())
{
	inContextSlider = null;

	RaiseInContextSliderMonitor raiseMonitor =
	sliderSelection.raiseMonitor();

	raiseMonitor.setInContextSliderActivator(this);

	if (!sliderSelection.isSelecting())
	{
		raiseMonitor.cancel();
		raiseMonitor.waitThenShow();
	}
	else
	{
		raiseMonitor.raiseWithoutDelay();
	}
```

This code first checks to see if an In-Context Slider should be activated by calling the method shouldActivate(). This method should be overridden by any classes wishing to have special cases for when and when not an In-Context Slider should be activated to layer 1. If shouldActivate() returns true, it then sets the In-Context Slider assigned to this activator to null. It then obtains a RaiseMonitor object that is responsibly for delaying the appearance of the layer 1 navel. It assigns the activator object to the RaiseMonitor object. In that way, the RaiseMonitor object knows where to display the activated In-Context Slider when necessary. The code then checks to see if multi-activation is currently occurring. If multi-activation is occurring, no delay is needed to show the navel, so the RaiseMonitor object is told to immediately show the navel. If multi-activation is not

occurring, the RaiseMonitor object by calling the method waitThenShow() is told to start a thread that waits for a pre-defined delay before showing the In-Context Slider.

A word in a text surrogate is represented by the class cf.visualize.TextChunk-TokenVisual. Words in details-on-demand are represented by two sub-classes of cf.gui.im.IMTextTokenGUIt. A word in a field value uses cf.gui.im.IMFieldValue-TextTokenGUIt, and a word in a field label uses cf.gui.im.IMFieldLabelTextTokenGUIt. IMTextTokenGUIt provides general functionality for these words as In-Context Slider activators. The more specific classes override the generic functionality when necessary and provide additional operation, such as adjusting the color of all non-stop words in a details-on-demand field when expressing interest in the entire field through the field label.

cf.visualize.Surrogate, the class that represents surrogates in combinFormation, implements the InContextSliderActivator interface. However, activation functions differently from the normal case. The SurrogateInContextSlider that is activated by a Surrogate is displayed along with details-on-demand, in-context tools, and the edit palette. Therefore, activation of the SurrogateInContextSlider is handled by the raise mechanism for all these in-context objects as opposed to the activate() method. As an InContextSliderActivator surrogate has an update() method that is called when the value of the corresponding InContextSlider is set. The update() method propagates expressed interest to the appropriate places (e.g. relevant terms in details-on-demand).

APPENDIX B

This appendix contains a collection of compositions created by participants in Experiment 2 using the In-Context interface.

B-1


Discover France!
Everything is Sexier in Paris
SEX
I'd Rather Be In PARIS
VOGUE
PRETTY ROMANTIC
Internships
French Fashion

B-2


being built, Nijo Castle, Seiryu- en Garden
PennDesign Home :: Architecture :: Study Abroad
STUDY ABROAD IN TOKYO LAKELAND
Asuka and Nara architecture
INSTITUTE FOR STUDY ABROAD PROGRAMS
Japan
Programs in Japan
Himeji Castle (built in its present form 1609), popularly known as White Heron Castle, with its gracefully curving roofs and its complex of three subsidiary towers around the main tenshu (or keep), is one of the most beautiful structures of the Momoyama period.
Japanese architecture (

B-3

VITA

Name	Andrew Webb
Permanent Address	Department of Computer Science Texas A&M University TAMU 3112 College Station, TX 77843-3112
Education	B.S. in Computer Science Texas A&M University College Station, TX 77843, 2004 M.S. in Computer Science Texas A&M University College Station, TX 77843
Professional Experience	Web Designer, Page-Up America, The Woodlands, TX, 2004-2007 Teaching Assistant for Structures of Interactive Information, Texas A&M University, Fall 2004.
Conference Papers	Kerne, A., Koh, E., Smith, S.M., Choi, H., Graeber, R., Webb., A., Promoting Emergence in Information Discovery by Representing Collections with Composition, *Proc ACM Creativity & Cognition*, Washington DC, June 2007. Webb, A., Kerne, A., Koh, E., Joshi, P., Park, Y., Graeber, R., Choreographic Buttons: Promoting Social Interaction through Human Movement and Clear Affordances, *Proc ACM Multimedia*, Santa Barbara, Oct 2006. Kerne, A., Koh, E., Dworaczyk, B., Mistrot, J.M., Choi, H., Smith, S., Graeber, R., Caruso, D., Webb, A., Hill, R., Albea, J., A Mixed-Initiative System for Representing Collections as Compositions of Image and Text Surrogates, *Proc Joint ACM/IEEE Conf. on Digital Libraries (JCDL)*, June 2006, Chapel Hill.