Model-Based Analysis of Required Knowledge for Successful Interaction With A Novel Display

Muneo Kitajima

Institute of Cognitive Science
University of Colorado
Boulder, Colorado 80309-0344

&

National Institute of Bioscience and Human-Technology
Agency of Industrial Science and Technology, MITI
1-1 Higashi Tsukuba Ibaraki 305, JAPAN

e-mail: kitajima@nibh.go.jp
WWW: http://www.aist.go.jp/NIBH/~b0544/

Abstract

The goal of this study is to identify the knowledge users need to discover correct actions in novel interfaces. Users' task performances are simulated with the Kitajima and Polson (1995) comprehension-based model of action planning for display-based human–computer interaction (HCI). The action planning model is provided with a sufficient set of interaction knowledge for performing a graphing task, and then simulates the task in various situations defined by the combination of the following two conditions. The first condition concerns the degree of specificity of goal representations. The model is provided with goals with varied levels of detail, including the representations of task goal and desired device state. The second condition concerns the number of screen objects that the model considers in the action selection processes. The model pays attention to either a limited number of task relevant screen objects or all the screen objects from the entire screen. Results demonstrate that specific representations of a desired device state are critical for correct action planning. However, even without these representations, which is likely in the use of novel interfaces, the action planning model can act correctly when a task goal representation closely matches the correct screen object's label and when the model focuses on the task relevant area of the screen.

1. Introduction

Kitajima and Polson's (1995) comprehension-based model of action planning describes skilled users' performance on graphically based interfaces. In their action planning model, skilled users were assumed to have sufficient knowledge to perform a task on a given interface, including knowledge for parsing the display and knowledge about interaction conventions. These skilled users were also assumed to have correct representations of the task and of the critical display states that appear while performing the sequence of actions for the task. The action planning model simulated skilled users' performances—performances characterized by rapid and accurate action selections. In addition, the model uncovered a mechanism that accounts for errors (i.e., action slips) committed by skilled users.

This article describes simulation experiments that serve as a step for extending Kitajima and Polson's (1995) action planning model to a model that describes users' performances in a broader HCI context. One goal is to develop a model that describes how experienced users learn novel interfaces. Therefore, in this study experienced users' performances on a task with a novel application interface display were simulated. I explored whether the action planning model could successfully perform tasks with vague, incomplete, or even missing task or device goals. The initial conjecture was that users of a new application would have some understanding of the tasks they were going to perform, and that they would be able to formulate more or less complete task goals. However, new users would not be able to formulate the precise device goals required because they had never interacted with the application. Specifically, the simulations used in this study model an experienced user of the Macintosh who had never used Cricket Graph. The model also incorporated knowledge of Macintosh interface conventions.

Goals play a crucial role in Kitajima and Polson's (1995) action planning model, in which the degree of closeness of a screen object to the current goals, as defined by the model, affects the result of the process that selects appropriate screen objects for the next action. Kitajima and Polson (1995) simulated tasks using the goals that had large closeness to the correct screen object for the next action. No errors were due to the goal representations.

By contrast, in this study the representations of goals are varied. Certain goal representations would cause an error in the form that the model fails to attend to the correct screen object. The next section explains how goal representations affect action planning processes by providing a review of Kitajima and Polson's (1995) comprehension-based model of action planning. Following the review, the simulation experiments and their results are presented. The results are then analyzed in detail by describing the mechanisms by which correct screen objects or incorrect ones were selected in the simulation experiments. Finally, this article provides a specification of required knowledge for successful interaction with a novel display by experienced users.

2. Action Planning Model of Display-Based Human–Computer Interaction

Kitajima and Polson's (1995) comprehension-based model of action planning of display-based human–computer interaction is a generalization of Mannes and Kintsch's (1991) theory of action planning. Mannes and Kintsch extended the construction–integration theory of text comprehension (Kintsch, 1988) to action planning in human–computer interaction. Their action planning model

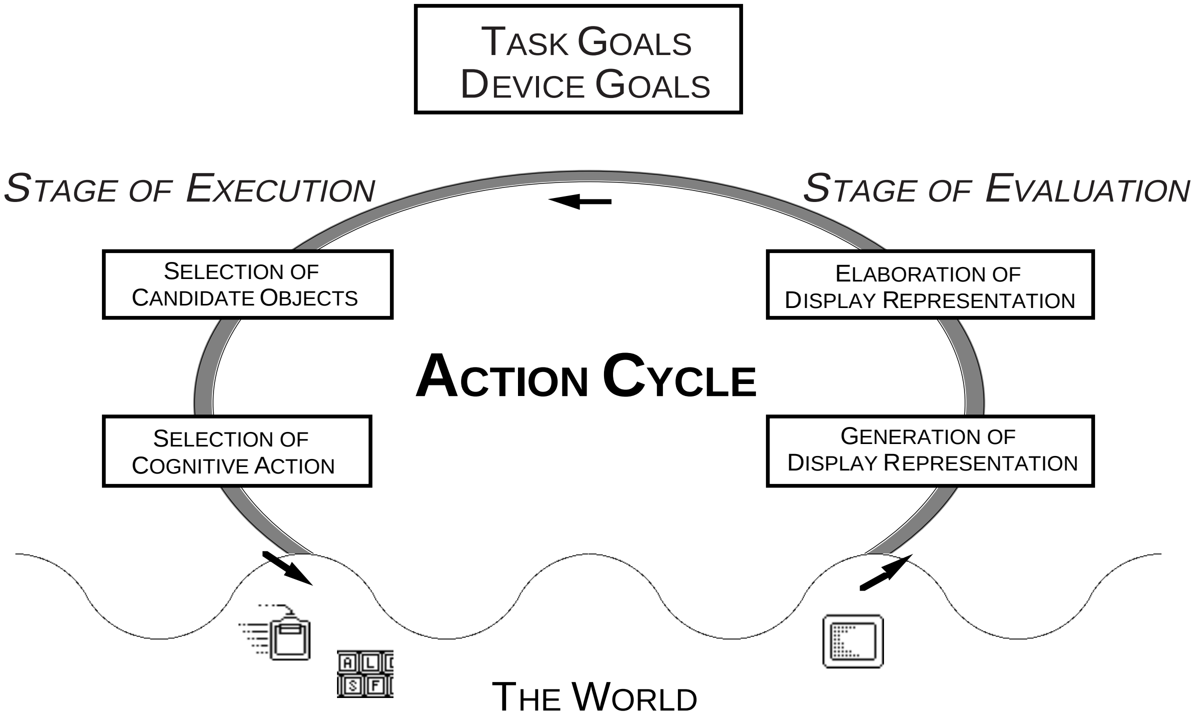
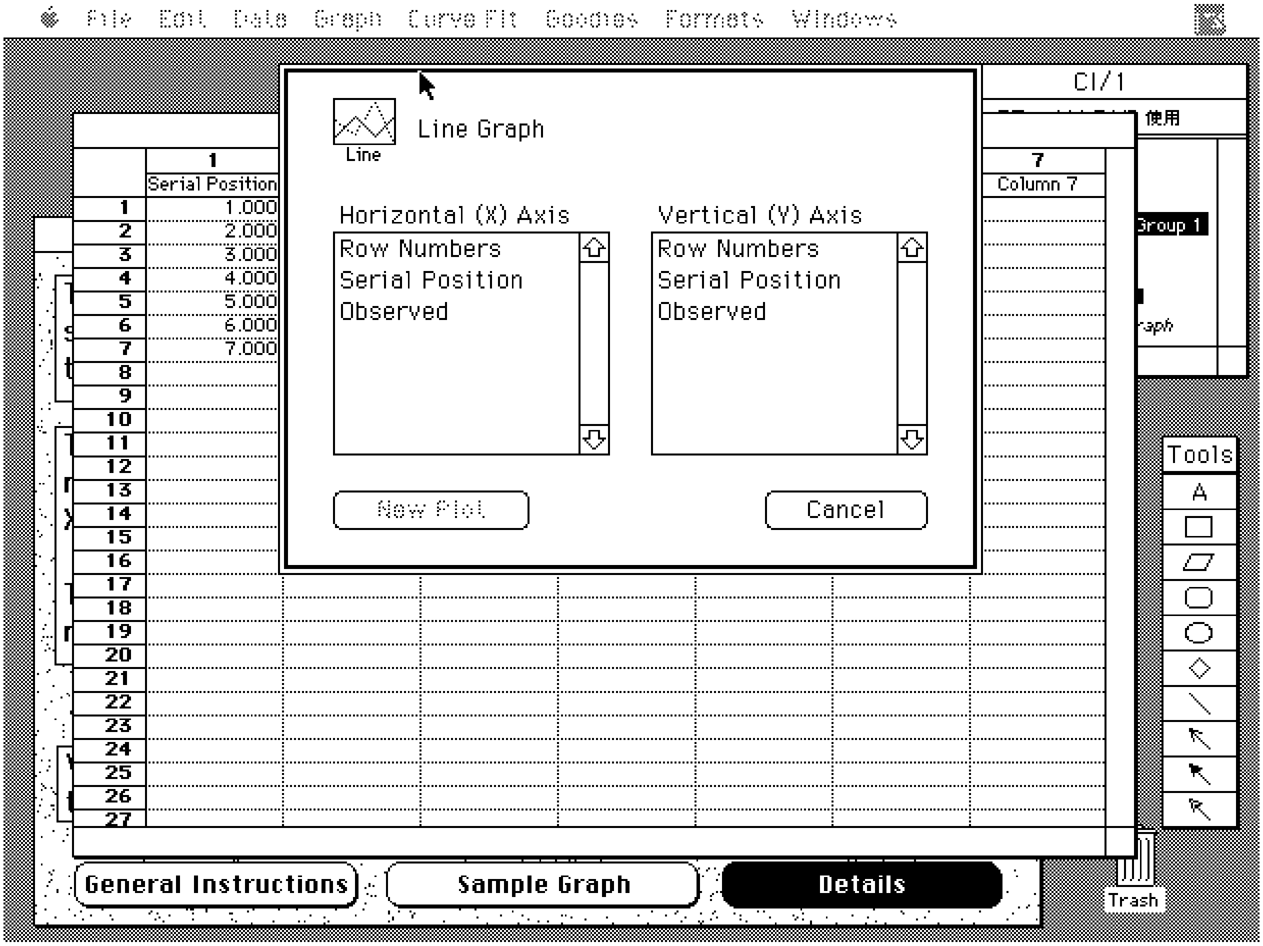


Figure 1 – The action planning model of correct performance and errors in skilled, display-based HCI (Kitajima & Polson, 1995).

took as input a representation of users' goals, a propositional representation of the text containing the task description, and a schematic representation of the task context. Their model generated the commands required to perform the task described in the text. Mannes and Kintsch argued that text comprehension and action planning can be conceived as similar tasks. Readers and planners must integrate their goals and information from diverse sources to select one of many alternative interpretations of a text or one out of many competing plans of action.

While Mannes and Kintsch's (1991) action planning model selects commands by comprehending the task described in text, Kitajima and Polson's (1995) action planning model selects actions, such as moving the mouse cursor and clicking the mouse button, by comprehending the task by integrating goals and the knowledge about interface conventions and meaning of screen objects. Like text comprehension, these action selections are guided by goals. Kitajima and Polson represented their action planning model as an evaluation–execution action cycle (see Figure 1). This action cycle framework is taken from Hutchins, Hollan, and Norman's (1986) analysis of direct manipulation. This framework describes action planning as a goal-driven process that evaluates the consequences of the last action and then generates the next action to be executed.

There are other important ideas integrated into Kitajima and Polson's (1995) action planning model; including the view about the nature of display-based human–computer interaction (e.g., Hutchins et al., 1986), theoretical ideas about the nature of display-based problem-solving (e.g., Larkin, 1989), action planning (Mannes & Kintsch, 1991), and task and device representations (Payne, Squibb, & Howes, 1990).



	1 Serial Position
1	1.000
2	2.000
3	3.000
4	4.000
5	5.000
6	6.000
7	7.000

Figure 2 – Example task.

2.1 Example Task

This section describes Kitajima and Polson’s (1995) action planning model by tracing its behavior during part of a task. This task involves drawing a line graph for data contained in a table with columns labeled with variable names. The example plotting task is:

> PLOT_TASK: plot numbers in the column labeled ‘Observed’ as a function of
> numbers in the column labeled ‘Serial-Position’

Users first select a graph type (e.g., line, bar, pie) from a pull-down menu, causing the dialog box in Figure 2 to appear. The column labels are displayed in two scrolling lists, and the label Serial-Position appears in both x- and y-axis scrolling lists. The dialog box partially occludes the table, but the column of numbers labeled Serial-Position is visible in the background. To plot Observed as a function of Serial-Position, users must click to highlight Serial-Position in the x-axis scrolling list and Observed in the y-axis scrolling list. Finally, users point at the button New Plot and single-click it.

2.2 Goals

Kitajima and Polson’s (1995) action planning model assumes that skilled users have a schematic representation of the task in the form of a hierarchical structure involving two kinds of goals: task goals and device goals (Payne, Squibb, & Howes, 1990). They assumed that each task goal is associated with one or more device goals. The device goals specify device states that must be achieved to satisfy an associated task goal. When the action planning model is provided with a new display, the model retrieves task and device goals. The task and device goals for the example task were:

TG: perform “*select* Serial-Position as the x-axis”
DG: realize “Serial-Position-in-X-List *is-highlighted*”

Note that the argument Serial-Position-in-X-List in DG refers to the screen object that appears as the second item in the scrolling list with the heading, Horizontal (X) Axis.

2.3 The Evaluation Stage

The action planning model is given a representation of a new display in the form of a large collection of screen objects; each screen object is described by several propositions. These propositions include only limited information about the identity of each object and its appearance, including visual attributes (e.g., color, highlighting).

The model simulates Hutchins et al.’s (1986) evaluation stage (shown in Figure 1) by elaborating this display representation with knowledge retrieved from long-term memory. The retrieval cues are the task and device goals and the propositions representing the current display. The probability that a cue retrieves a particular proposition representing a piece of knowledge in long-term memory in a single retrieval trial is proportional to the strength of the link between them. The propositions in long-term memory represent knowledge about the screen objects. For example, for the screen object Serial-Position-in-X-List the following knowledge items are represented in long-term memory:

Serial-Position-in-X-List *has-label* Serial-Position
Serial-Position-in-X-List *is-a-member-of* Line-Graph-Dialog-Box
Serial-Position-in-X-List *can-be-pointed-at*
Serial-Position-in-X-List *can-be-selected*

The result of the display evaluation is represented as a collection of propositions representing the display and the knowledge retrieved from long-term memory. This representation is called the *elaborated display representation*. The elaboration process is stochastic and is taken from Kintsch (1988). The number of memory retrieval trails is a model parameter, called the *elaboration parameter*. Kitajima and Polson (1995) discussed in detail the predictions and implications that follow from this stochastic elaboration process.

2.4 The Execution Stage

The execution stage of Hutchins et al.'s framework (1986) is modeled by two construction–integration cycles. The first construction–integration cycle selects three screen objects as possible candidates for the next action. An important feature of Kitajima and Polson's (1995) action planning model is that the display representation is a detailed description of an actual large format display. Thus, the model's display representation can incorporate up to 100 screen objects. All screen objects are candidates for possible actions. During the initial construction phase, all screen objects are represented as respective *candidate object nodes* and combined with the goals and the elaborated display representation to construct the network. When the integration process converges, the model selects the three most highly activated candidate object nodes. These nodes define three screen objects that are candidates for the next action.

This initial integration process is dominated by two factors:

1) strong links from the goals to propositions in the network that share arguments with the goals (the weight assigned to these links is a model parameter, called the *goal magnification factor*)
2) the number of propositions necessary to link goals to screen objects

As a result, the action planning model selects screen objects that are related closely to the task and device goals as candidates for the next action. Device goals can directly specify a screen object, and, thus, can be directly linked to the screen object represented in the network. The argument, Serial-Position-in-X-List in DG, is such an example. Task goals can be linked to screen objects through labels. This linkage is indirect. For example, the task goal defined by TG is linked to the screen object, Serial-Position-in-X-List, through the two labels in TG (i.e., Serial-Position and x-axis). TG is linked to propositions representing display elaboration that share these labels. Some of these propositions are linked to the candidate object node representing the correct screen object, Serial-Position-in-X-List. Thus, the amount of activation that the correct candidate object node collects is significantly dependent on the detailed structure of the network.

The second construction–integration cycle selects an action to be performed on one of the three candidate objects. During the construction phase of this second cycle the model generates a network with representations of all possible actions on each candidate object. Examples would include single-clicking Serial-Position-in-X-List and moving Serial-Position-in-X-List. At the end of the second integration phase, the action planning model selects the most highly activated object–action pair as the next action to be executed. The process is dominated by the same two factors described above. However, the relevant interaction knowledge must be retrieved during the evaluation stage. For example, the action planning model must retrieve the fact that objects in the scrolling list can be selected.

3. Simulation of Experienced Users' Performances on a Novel Interface

Kitajima and Polson (1995) assumed that they were modeling skilled users of the graphing package, and they gave the action planning model the explicit goals required, examples of which are shown in TG and DG. This section explores whether the model could successfully perform tasks with vague, incomplete, or even missing goals.

3.1 Task

The simulation focuses on part of the first subtask in the graphing task, PLOT_TASK. This subtask involves specifying a graph type (e.g., line or pie) and then interacting with a variables selection dialog box to specify what columns in the spreadsheet contain the values for the x- and y-axes. The dialog box is shown in Figure 2. The current simulation experiments started with a display representation defined by Figure 2. The correct action sequence involves moving the mouse cursor to point at the screen object, Serial-Position-in-X-List, followed by single-clicking on that object. As we shall see, TG and DG define the specific version of goals required to generate the correct actions.

3.2 Model Parameters

Kitajima and Polson (1995) reported the ranges of parameters necessary for the action planning model to simulate rapid and correct action selection. In their simulations, the model was provided with correct representations of task and device goals for a fixed display representation. The following were the major results:

1. Large values of the argument overlap weight, 4.0, which defines a base link strength given to any pair of propositions that share arguments, are necessary to make the network converge fast in the network integration processes (i.e., rapid action selection).
2. Large values of the goal magnification factors, 16.0, are necessary for the model to make a correct action selection by selective activation of the correct candidate object node in the candidate objects selection process, and of the correct object–action pair node in the action selection process.
3. Large numbers of the elaboration parameter are necessary in the display elaboration process to avoid errors resulting from failure of retrieving critical knowledge for correct action selection from long-term memory.

Based on these findings, the current simulation experiments used the same parameters for the argument overlap weight (4.0) and the goal magnification factor (16.0), as well as a significantly large value of the elaboration parameter (24). This parameter set assures correct action selections if correct goals are specified.

3.3 Simulation Conditions

In the simulation experiments, the action planning model was provided with varied sets of representations of task and device goals and the display contents. There were three versions of the task goals, two versions of the device goals, and two conditions for screen objects to be included in the simulations.

Three variations of the task goal were:

Specific task goal: perform “*select* Serial-Position *as* x-axis” ; *same as* TG
Redundant task goal: perform “*select* Serial-Position *as* x-axis *from* Line-Graph-Dialog-Box”
No task goal: perform “*nil*”

The representation of the redundant task goal has the extra argument, Line-Graph-Dialog-Box, compared to the specific task goal. Line-Graph-Dialog-Box refers to the entire dialog box. As we shall see, this extra argument has advantages and disadvantages for action planning. It may help the model focus on the dialog box, but the argument may also blur the model’s attention after it has successfully concentrated on the dialog box.

There were two versions of the device goals:

Specific device goal: realize “Serial-Position-in-X-List *is-highlighted*” ; *same as* DG
No device goal: realize “*nil*”

The specific device goal represents the desired device state when the correct action is performed. It is unlikely that a user of a new application would have such specialized knowledge—that knowledge would be acquired through successful experiences. However, this goal is included in this simulation to examine the effect of such specialized knowledge.

Two variations were assumed for the representations of the display contents in terms of number:

Whole: 78 screen objects in the whole screen shown in Figure 2
Part: 9 screen objects from only the dialog box

The categories of the screen objects and their numbers are listed in Table 1.

Table 1 - Screen objects used in the simulation.

Location in Display	**Condition**	
	whole(78)	**part(9)**
Line Graph dialog box	**11**	**9**
Title: “Line Graph”	1	1
Horizontal (X) Axis Selection List: title, Row Numbers, Serial Position, Observed	4	4
Vertical (Y) Axis Selection List: title, Row Numbers, Serial Position, Observed	4	4
Radio Buttons: New Plot, Cancel	2	0
Desktop	**2**	**0**
desktop itself, Trash	2	0
Desktop Menu	**11**	**0**
Finder Menu: apple, help, application	3	0
Cricket Graph Menu: File, Edit, Data, Graph, Curve Fit, Goodies, Formats, Windows	8	0
Tools	**11**	**0**
title, text, etc.	11	0
Spreadsheet	**39**	**0**
title, close box, zoom box	3	0
Rows:	25	0
Columns: column 1 and 7, others are covered by the dialog box	2	0
Column Labels: Serial Position, Column 7	2	0
Data in Column:	7	0
CI/1 Window	**1**	**0**
title	1	0
Hyper Card Window	**3**	**0**
Buttons:	3	0

3.4 Simulation Procedure

Action selections for each of the possible combinations were simulated. However, meaningless combinations like no task goal and no device goal were excluded. The simulation procedure is the same as the one reported in Kitajima and Polson (1995). For each combination of conditions, the program first read corresponding representations of the task goal, the device goal, and the display from an input file. The mouse cursor initially pointed at the desktop (shown in Figure 2). Then the program simulated the action planning process assumed by Kitajima and Polson (1995).

The simulation of the first action selection included display elaboration, object selection, and object–action pair selection. The simulations were done with a large value of the elaboration parameter. Therefore, the model retrieved all relevant knowledge about each screen object with a high probability and incorporated this knowledge into the network necessary to perform any legal actions for any screen objects. In the current simulations, errors due to incomplete display elaboration (i.e., action slips in Kitajima and Polson's, 1995, terminology) were least probable.

In the object selection process for the first action, the program chose the three most highly activated candidate object nodes. These candidate objects were carried over to the object–action pair selection process. Then, the program selected the most highly activated object–action pair node as the first action. This action was usually moving the mouse cursor to one of the three candidate objects.

During the simulation of the second action, the model executed only the object selection and object–action pair selection processes, because moving the mouse cursor does not lead to a major display change that triggers the elaboration process. The model almost always performed an action on the object pointed at by the mouse cursor. The typical action was single-click. Thus, if the model moved the mouse cursor to the correct scrolling list item, Serial-Position-in-X-List, it would correctly select it by single-clicking.

4. Results and Analyses

As described in the previous section, the simulation typically single-clicked the object pointed at by the mouse cursor. The action of moving the mouse cursor to one of the three candidate objects is the result of the first action selection. In addition, the first action was performed on the primary candidate object, which refers to the screen object associated with the most highly activated candidate object node in the first object selection process. Thus, the whole course of action selections was determined by the results of the object selection process for the first action selection. Thus, in the following, the results and analyses are presented by using the representation of the network constructed and integrated in the first object selection process.

The model's major source of difficulty was multiple objects with the same label in each of the two display conditions. There were two instances of the label Serial-Position when the model focused on the dialog box (i.e., part condition) and three instances when the model was given the complete representation of the display (i.e., whole condition; see Figure 2).

The competing three screen objects were represented in the network as follows:

Prop 1	Serial-Position-in-X-List *is-on-screen*
Prop 2	Serial-Position-in-Y-List *is-on-screen*
Prop 3	Column-Label-1 *is-on-screen*

The fact that they were all labeled as Serial-Position is represented as follows:

Prop 4	Serial-Position-in-X-List *has-label* Serial-Position
Prop 5	Serial-Position-in-Y-List *has-label* Serial-Position
Prop 6	Column-Label-1 *has-label* Serial-Position

Respective candidate nodes were:

Prop 7	Serial-Position-in-X-List *is-candidate-for-next-action*
Prop 8	Serial-Position-in-Y-List *is-candidate-for-next-action*
Prop 9	Column-Label-1 *is-candidate-for-next-action*

In each simulation of the object selection process for the first action selection, the constructed network included props 1 through 9 along with the representations of the display defined by the corresponding display condition and its elaboration. In the integration process, the candidate object nodes (including Prop 7, 8, and 9) were activated. As the result, the model chose the three most highly activated candidate object nodes out of all the candidate object nodes. As described previously, the primary candidate object was the screen object where pointing and single-clicking actions were performed in the succeeding processes in the simulation. The question is what simulation condition lead to preferable activation of the correct candidate object node (i.e., Prop 7).

4.1 Specific Device Goal (DG)

If the model was given the specific device goal, DG, then there was a direct link between the correct screen object, Serial-Position-in-X-List, and the goal. DG linked directly to Prop 7. For these conditions in the simulations, the model always made the correct response because the ambiguity of multiple objects with the same label was resolved by the device goal.

Table 2 shows the activation values of the candidate object nodes representing these screen objects in the simulation conditions in which the specific device goal was present. From Table 2, we can see the powerful effect of the existence of the specific device goal on the correct action selection. The activation values for the correct candidate object node were almost a factor larger than those of the competitors because the correct candidate object node had a direct link to the specific device goal. This device goal is a powerful activation source via the argument, Serial-Position-in-X-List .

Table 2 – Activation values of candidate object nodes when DG is present.

Condition		Activation values of candidate object nodes		
Screen	Task Goal	Serial-Position-in-X-List	Serial-Position-in-Y-List	Column-Label-1
Part	Specific	**0.9209**	0.1010	N/A
	Redundant	**0.5185**	0.0617	N/A
Whole	Specific	**0.5340**	0.0486	0.0847
	Redundant	**0.5501**	0.0655	0.0799

4.2 No Device Goal

When no device goal is provided, as in the case of in a novel application, the task goal must resolve the ambiguity caused by multiple identical labels. As we shall see, the degree of specificity is a critical determinant of successful interaction when a device goal is not present.

4.2.1 Specific Task Goal / Part Display Condition

On part condition, where the model's attention was restricted to the dialog box, the specific task goal TG worked because TG is linked to the correct candidate object node, Prop 7. The labels in TG, Serial-Position and x-axis, defined activation paths to Prop 7. These paths enabled the model to activate more strongly the correct candidate object node. As shown in Table 3, the simulation could select the correct object on the specific task goal condition. The activation value of the correct candidate object node representing Serial-Position-in-X-List, Prop 7, was slightly greater than that of Serial-Position-in-Y-List, Prop 8. Another 20 simulation runs for this experimental condition were conduced. They confirmed that these activation values were constant because the constituents of the network and the weights of links of the network were constant in this experimental condition. The elaboration process carried out exhaustive knowledge activation, causing the constant network in the selection processes.

In the following, this section examines the mechanism by which the correct candidate object node got more activation than the incorrect one. This mechanism concerns the activation paths that link TG and the correct candidate object node Prop 7. In the case of specific task goal, TG, these paths are illustrated in Figure 3. Among these paths, a path shown by solid lines plays a critical role on selective activation of the correct candidate object node Prop 7. The argument, x-axis, in TG establishes this path to activate the correct candidate object node by using the following propositions. This knowledge was retrieved in the elaboration process.

Prop 10	Horizontal-X-List *has-label* Horizontal x-axis
Prop 11	Horizontal-X-List *has* Serial-Position-in-X-List

Horizontal-X-List refers to the whole scrolling list labeled by "Horizontal (X) Axis" in Figure 2. The argument x-axis in TG overlaps with Prop 10, which links to Prop 11 using the overlapping argument, Horizontal-X-List, and then to the correct candidate object node Prop 7 via Serial-Position-in-X-List. Due to this path, the model activated the correct candidate object node, Prop 7, more strongly than the wrong one, Prop 8. However, the other argument Serial-Position in TG equally activated Prop 7 and Prop 8.

Table 3 – Activation values of candidate object nodes when the specific device goal is not present.

Condition		Activation values of candidate object nodes		
Screen	Task Goal	Serial-Position-in-X-List	Serial-Position-in-Y-List	Column-Label-1
Part	Specific	**0.2224**	0.2150	N/A
	Redundant	0.0921	**0.0958**	N/A

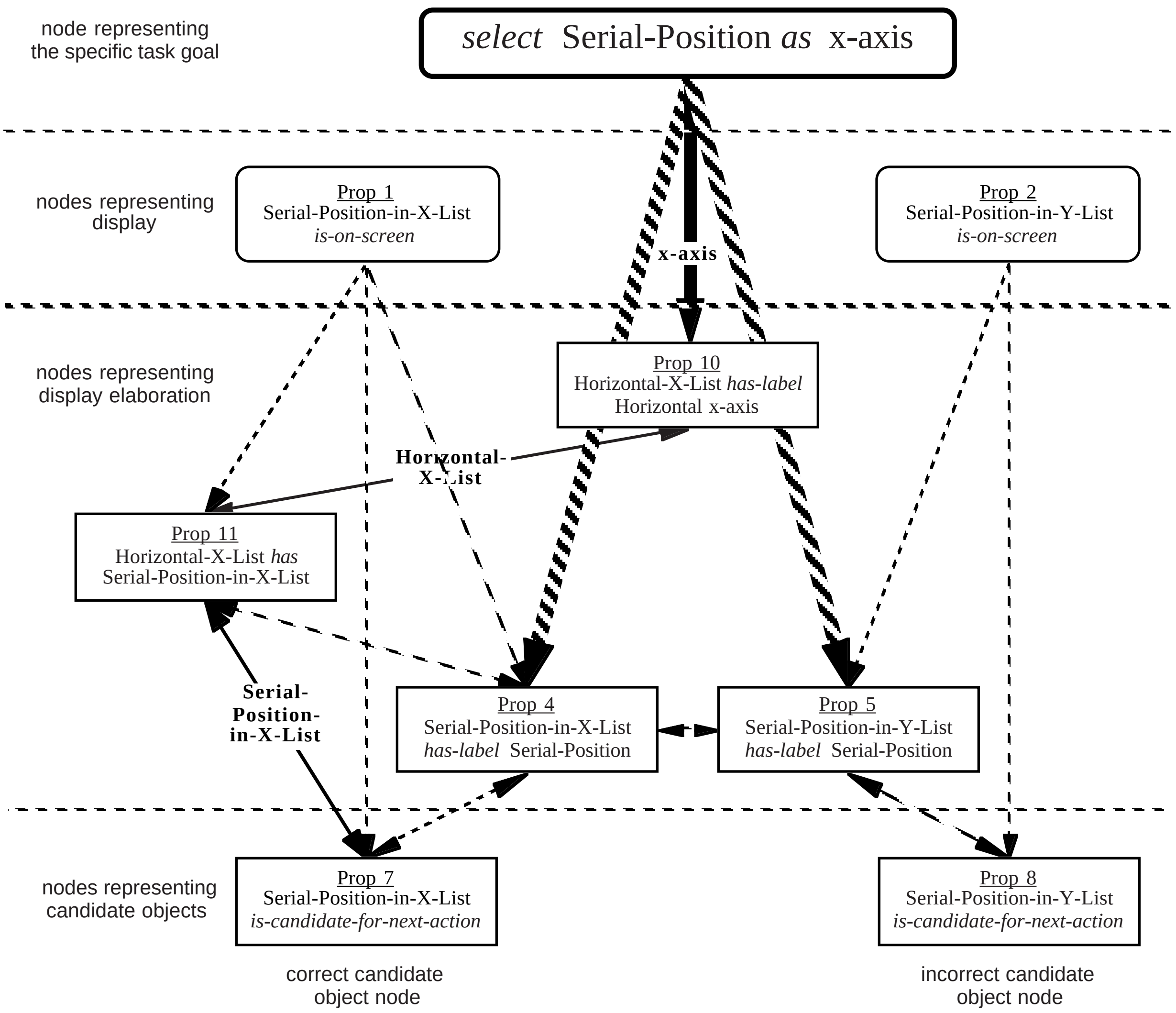


Figure 3 – Activation paths to the correct and the wrong candidate nodes with the specific task goal.

4.2.2 Redundant Task Goal / Part Display Condition

Even in the part condition, the model's behavior on the redundant task goal condition was somewhat different from the one on the specific goal condition. The redundant task goal did not always generate correct actions. The redundant argument Line-Graph-Dialog-Box linked to other screen objects. These links avoided the model from focused activation of the correct candidate object node as in the case of the specific goal condition shown in Figure 3.

The extra argument in the task goal representation, Line-Graph-Dialog-Box, overlaps the following 10 propositions in long-term memory. These proposition define identification of items in the dialog box.

Prop 12 Horizontal-X-List *is-a-member-of* Line-Graph-Dialog-Box
Prop 13 Row-Numbers-in-X-List *is-a-member-of* Line-Graph-Dialog-Box
Prop 14 Serial-Position-in-X-List *is-a-member-of* Line-Graph-Dialog-Box
Prop 15 Observed-in-X-List *is-a-member-of* Line-Graph-Dialog-Box

Prop 16 Vertical-Y-List *is-a-member-of* Line-Graph-Dialog-Box
Prop 17 Row-Numbers-in-Y-List *is-a-member-of* Line-Graph-Dialog-Box
Prop 18 Serial-Position-in-Y-List *is-a-member-of* Line-Graph-Dialog-Box
Prop 19 Observed-in-Y-List *is-a-member-of* Line-Graph-Dialog-Box

Prop 20 New-Plot-Button *is-a-member-of* Line-Graph-Dialog-Box
Prop 21 Cancel-Button *is-a-member-of* Line-Graph-Dialog-Box

When the cue in the redundant task goal, Line-Graph-Dialog-Box, was used successfully to activate some of these propositions in the elaboration process, the model increased the weight of the links between the redundant task goal and the retrieved proposition by a certain amount as defined by the Kitajima and Polson (1995) action planning model. Because the process is stochastic, the resultant network reflected statistical fluctuations and, thus, the pattern of activation values resulting from the integration process would vary over simulation trials.

Figure 4 shows the activation values of Prop 7 and Prop 8 in 20 simulation runs under this experimental condition. The model selected the correct object 12 times. These trials are marked on the horizontal axis. For the other eight times, the model activated the candidate object node Prop 8

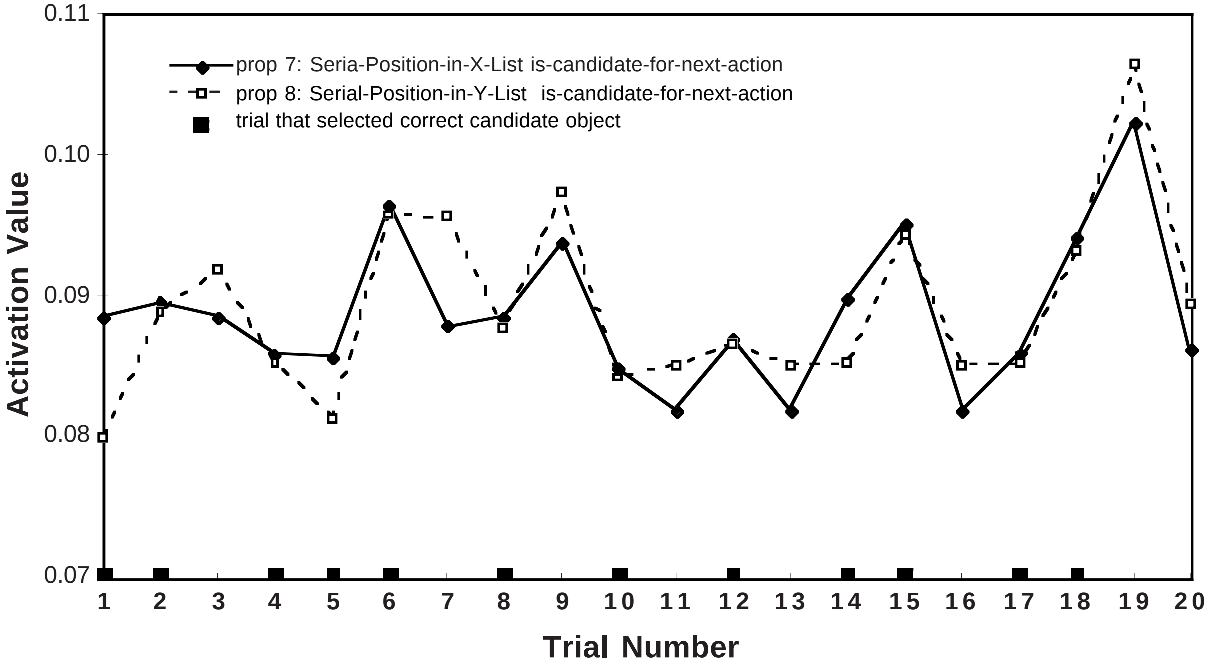


Figure 4 – Candidate object selection with the redundant task goal, no device goal, and part display condition. The simulation trials that selected the correct object are marked.

corresponding to Serial-Position-in-Y-List most strongly. This is due to the results of the construction process that assigned different link strengths depending on the results of the elaboration process. Even though the constituents of the network were not changed over the simulations, the model formed networks with different link assignments depending on the propositions that the redundant task goal activated.

4.2.3 Whole Display Condition

In the cases where no device goal was given and the model did not restrict its attention to the relevant part of the display, the model always selected Column-Label-1 as the next object (shown in Table 4). The mere fact that the number of objects identified as a member of the spreadsheet is large is the direct cause of the model's wrong selection. The correct candidate object node did gain extra activation from the specific or redundant task goal. However, the activation that the wrong candidate object node for Column-Label-1, Prop 9, obtained was greater than that of the correct candidate object node due to a so-called "set effect." Recall that membership of an object is represented like "Observed-in-X-List *is-a-member-of* Line-Graph-Dialog-Box." There were 10 members in the dialog box. On the other hand, there were 39 members in the spreadsheet (as shown in Table 1). The members were mutually linked, and, thus, they activated each other in the integration process of the object selection process. The competition was the focused activation of the correct candidate object node through additional information specifying the current task (i.e., x-axis and Serial-Position, and Line-Graph-Dialog-Box), and the collective activation through the set effect working among the spreadsheet members. The results of the simulation show that the set effect was stronger, at least in the simulated cases, than the additional information characterizing the task goal.

Table 4 – Activation values of candidate object nodes when the specific device goal is not present, and the model considered all the screen objects.

Condition		Activation values of candidate object nodes		
Screen	Task Goal	Serial-Position-in-X-List	Serial-Position-in-Y-List	Column-Label-1
Whole	Specific	0.0482	0.0478	**0.0869**
	Redundant	0.0648	0.0672	**0.0818**

The activation value of the correct candidate object node would have become larger than that of the candidate object node for Column-Label-1 if more information had been specified in the task goal, and / or less members from the spreadsheet had been attended. This prediction is derived from a comparison of the rows in Table 4. However, the additional information in the goal representation might not help the model select the correct one from the two Serial-Positions in the dialog box. Rather, the additional information in the goal would establish paths that do not selectively activate the correct candidate object node as in the case of the redundant task goal.

5. Conclusion

This study showed that Kitajima and Polson’s (1995) action planning model can reliably generate the correct action sequence with no device goal. Thus, the action planning model could perform a task with a new program. However, there are conditions for this to happen: users must have specific task goals and they must restrict their attention to the task relevant screen area.

The task goal must be stated exactly as given by the specific task goal, like TG. An apparently reasonable task goal like the redundant task goal did not work. The redundancy of the goal caused networks with various link assignments. The detailed pattern of the links from the task goal has a significant effect on the model’s object selection process, and, thus, the results of the object selection process in the case of the redundant task goal suffered from stochastic fluctuations. Furthermore, an unanalyzed task goal like “Plot Observed as a function of Serial-Position” will not work either. In summary, the task goal had to be directly linked to the labels for the x-axis scrolling list, Horizontal (X) Axis, and for the correct screen object in that scrolling list, Serial-Position-in-X-List. In general, paths that link task goals and the correct candidate object are indirect. Thus, the task goal must be specific enough, like TG, for the model to make focused activation of the correct candidate object node. Otherwise, the model may select an incorrect object for next action.

Furthermore, there was a significant effect of number of screen objects on the results of the object selection process. If the model was required to make the correct actions with a screen representation that included both the dialog box and the data table in the background with the distracting label Serial-Position, these additional distracting objects prevented the action planning model from successfully generating the correct action sequence. However, limiting the focus of attention to the nine screen objects defined by the dialog box enabled the specific task goal to generate the correct action sequence.

These conditions for successful action selection cannot be controlled by Kitajima and Polson’s (1995) action planning model itself. This situation is the same for the other models in HCI. In fact, the models in HCI dealing with either performance, including CCT (Kieras & Polson, 1985) and GOMS (Card, Moran, & Newell, 1983) or competence, such as TAG (Payne & Green, 1986) and D-TAG (Howes & Payne, 1990), consider tasks as given or represented in an ad hoc way. The conditions for successful action selection specified in this article define issues to be considered in developing models in display-based HCI that deal with more realistic situations than the current models consider (e.g., situations in which tasks are given to users as instructions, or users interact with cluttered displays with multiple windows). Obviously, in these situations models deal with processes for generating workable goals and for attending to the task relevant screen area. One of these issues, how the goals are generated, is currently challenged by the author and his collaborators (Kitajima & Polson, 1996a; Tewillinger & Polson, 1996). A model of the goal generation process and its simulations are forthcoming (Kitajima and Polson, 1996b).

6. References

Card, S.K., Moran, T.P., & Newell, A. (1983). *The psychology of human–computer interaction*. Hillsdale, NJ: Lawrence Erlbaum.

Howes, A., & Payne, S.J. (1990). Display-based competence: Toward user models for menu-driven interfaces. *International Journal of Man–Machine Studies*, **33**, 637–655.

Hutchins, E.L., Hollan, J.D., & Norman, D.A. (1986). Direct manipulation interfaces. In D.A. Norman & S.W. Draper (Eds.), *User centered system design* (pp. 87–124). Hillsdale, NJ: Lawrence Erlbaum.

Kieras, D.E., & Polson, P.G. (1985). An approach to the formal analysis of user complexity. *International Journal of Man–Machine Studies*, **22**, 365–394.

Kintsch, W. (1988). The role of knowledge in discourse comprehension: A construction–integration model. *Psychological Review*, **95**, 163–182.

Kitajima, M., & Polson, P.G. (1995). A comprehension-based model of correct performance and errors in skilled, display-based human–computer interaction. *International Journal of Human–Computer Systems*, **43**, 65–99.

Kitajima, M., & Polson, P.G. (1996a). A comprehension-based model of exploration, in *Proc. CHI'96 Human Factors in Computer Systems*, 324-331, ACM.

Kitajima, M., & Polson, P.G. (1996b). A comprehension-based model of exploration. *ICS Technical Report*, **96-02**.

Larkin, J.H. (1989). Display-based problem solving. In D. Klahr & K. Kotovsky (Eds.), *Complex information processing: The impact of Herbert A. Simon* (pp. 319–342). Hillsdale, NJ: Lawrence Erlbaum.

Mannes, S.M., & Kintsch, W. (1991). Routine computing tasks: Planning as understanding. *Cognitive Science*, **15**, 305–342.

Payne, S.J., & Green, T.R.G. (1986). Task-action grammars: A model of the mental representation of task languages. *Human–Computer Interaction*, **2**, 93–133.

Payne, S.J., Squibb, H.R., & Howes, A. (1990). The nature of device models: The yoked state hypothesis and some experiments with text editors. *Human–Computer Interaction*, **5**, 415–444.

Tewillinger, R.B., & Polson, P.G. (1996). Task elaboration or label following: an empirical study of representation in human-computer interaction. In *Human Factors in Computing Systems CHI'96 Conference Companion*, 201-202.