No. **652**

HyperScaling : An Interactive Multimedia Data-Collection Tool

by

Noriyuki Matsuda

February 1996

HyperScaling : An Interactive Multimedia Data-Collection Tool

Abstract

The recent proliferation of object-oriented, visual programing platforms has opened a new stage for psychological experiments. Our HyperScaling has been developed in order to offer the basic data collection tool by incorporating the ideas of hypertext-type information linkages in multimedia, contingent arrangement of tasks and measurement, and interval or triple-value measurement of vague perceptions. HyperCard has been adopted to demonstrate the practical feasibility of the HyperScaling principles by virtue of its primitive but powerful visual object-orientedness. It has been successfully implemented in our real studies on consumer images and automotive risk perceptions. HyperScaling in not only limited to the quantification of judgments, but its principles are applicable to classification and matching tasks as well.

HyperScaling: An Interactive Multimedia Data-Collection Tool

The recent proliferation of object-oriented visual programming platforms such as HyperCard (Apple Computer, Inc.), SuperCard (Allegiant Technologies, Inc.), VisualBasic (MicroSoft, Inc.) and Visual C++, has opened a new stage for psychological measurement, particularly in the area of judgment and decision making. Researchers can now trace processes of how their subjects interact with the stimuli and other visually presented objects, rather than simply analyzing the end-responses of their subjects.

Another advantage accruing from the interactiveness is the possibility for designing a contingent experiment such that the task situation in one stage reflects responses in the previous stages or some external data like those of a negotiating party in a game-theoretic task. For the sake of simplicity, however, the present paper, will be confined to the stand-alone use of our data-collection software called *HyperScaling* the versatility of which increases when graphic and/or audio information is incorporated with text-based information.

As the prefix *Hyper* suggests, the central feature of the software is the information linkage that falls in two types: the Hypertext-linkage and the contingent measurement. The former, the hypertext linkage is widely implemented and enjoyed today on numerous WWW (World Wide Web) pages. Although not readily seen to a viewer, a hypertext is a directed graph of audio, graphic and text information (Nielsen, 1993) which takes two forms in our software. In the limited form, only a directly linked node(s) appears in the same or separate window(s). A subject may retrieve a detailed explanation about a stimulus by generating the predetermined event such as a mouse-click on the designated object (see Matsuda & Namatame, 1995; Matsuda & Shinoda, 1995; Matsuda, Shinoda & Takemura, 1995). This form will be particularly useful for studying the role of auxiliary information, for example, in the consumer images.

In the less limited form, a new page or window overrides the old one to present a directly connected node and its accompanying nodes like the ordinary transition of WWW pages. The most basic application of the form is the task transition upon the completion of one stage or at the request of a subject. If

necessary, the transition can be made contingent on the responses of the previous stage. The contingent transition is not unrelated to, but should not be confused with the contingent measurement to be explained below.

In making judgments and decisions, people often employ the anchoring-adjustment heuristic (Kahneman, Tversky & Slovic, 1982), adjusting an initially-held value, i. e., an anchor, until they reach a reasonable one. Sometimes the anchor spontaneously occurs to them as revealed by post-experimental interviews in attitude surveys. At other times it is externally offered by an experimenter as a reference value (Tversky, 1994). Whether anchoring results in a judgmental bias (Kahneman, Tversky & Slovic, 1982; Kahneman, 1992) or aids judgments, the adjustment process deserves an psychological investigation to which HyperScaling offers great advantage over the conventional paper-pencil method.

HyperScaling can deal with both internal and external anchors that are respectively linked to the subject's own earlier response(s) and to an external source such as an almanac, a field expertise and the opponent of a game. Technically speaking, anchors are treated as default values regardless of the content. An interesting application is seen in the analysis of hierarchically related consumer-product images by Matsuda and Namatame (1995).

The other form of the contingent measurement pertains to the uncertain perceptions about the state of affairs in general arising either from the imperfect knowledge or from the awareness of the natural variabilities (Matsuda, Shinoda & Takemura, 1995). The price of a car will illustrate the point. Usually there are various models sold at different prices under a single brand, e.g., the Ford Taurus. In addition to this within-brand variability, there are within-model price differences due to the options a buyer chooses to mount, the regional market condition and the like. Even with full awareness of such variabilities, very few would be able to specify the entire distribution, since it requires enormous amount of information. People, nevertheless, would be able to indicate the variability in terms of an interval (more specifically, lower- and upper-ends of an interval) and the most likely (or representative) value within it. The belief has led us to develop a triple-value method in our software (see, for early similar attempts, Hesketh, Pryor, Gleitzman & Hesketh, 1988; Rosa & Humphreys, 1988).

Provided that our triple-value measurement is congenial to human judgments

in general, we should also note the potential mental strain that a subject undergoes in the triple-value measurement. One can easily imagine the situation by moving the indicator objects for the lower and upper interval ends and the most likely value which are subject to the following constraint:

lower end $\leq$ most likely value $\leq$ upper end.

It would be too much to ask the subject to observe the constraint all the time, whereas a rigid control by program would hamper the usability. As a solution, HyperScaling offers the transitional method, on one hand, and the software restoration for a violation of the constraint, one the other. In the transitional method, the subject first complete the point evaluation, then move to the triple-value evaluations. Alternatively, the task may be set to start with an interval evaluation ensued by the triple-value evaluation. In the interval and the triple-value evaluations, whether the method is transitional or not, the user may slide an object in violation of the rule, since the software will relocate the objects when the sliding event is finished. It will be made clear in the next section that the restoration is a variant of the contingent measurement.

All the foregoing ideas are, in a sense, conceptual and technical extensions of the conventional scaling methods on a uni-dimensional scale. Before closing this section, we should refer to a two-dimensional method, a unique feature in HyperScaling. Briefly stated, its purpose is to capture the interrelated judgments. At the same time, it offers the ease of expression to subjects. It is our conviction that responses to multiple items are hardly independent from each other in practice. The two-dimensional method has been implemented in hope to help subjects consider two conceptually related items simultaneously by operating a single object rather than two objects on separate scales. Further merits will be presented in discussion of this paper.

The Design of HyperScaling

HyperScaling is language-independent in principle. Hence, an interested reader should be able to implement the idea in any object-oriented, visual-programing platform. The present system, nonetheless, has been developed in Macintosh HyperCard chiefly because of its simple but powerful object and language structure coupled with the versatile audio-visual presentation capacity. In designing the software, a particular emphasis has been placed on the portability to suit various experimental settings as well as other professional use. The primitive object-orientedness of HyperCard helps realizing portability in terms of modularity of program scripts and stack objects. Before we explain HyperScaling, a brief exposition to HyperCard seems in order.

HyperCard objects and the message path

The object-oriented programing is a software paradigm to cope with the growing complexity in the information management. The paradigm rests on the four major notions: abstraction, encapsulation, modularity and hierarchy (see Booch, 1991). The stack/card/background metaphor is the principle abstraction of information processing and display in HyperCard. A card and a background usually have one or more than one button or field parts where data and a program script are encapsulated. The message-path hierarchy makes it possible to place a modularized segment of a script in a higher object in the path. Interactive use of HyperCard exploits the message generation by an event-provoking action on an object to be explained more fully below. Those who are already familiar with HyperCard may skip the following explanations.

Insert Figure 1 about here

Shown in Figure 1 is the classification of the major objects of HyperCard. The attribute-slots of the topmost `Object`, i.e., names, physical properties, message generation and so forth, are shared by all the subordinate objects that fall into two groups depending on whether a script is attachable or not. (In fact, the `Object` belongs to the OS.) A script is a modularized set of statements which is roughly analogous to any meaningful set of subroutines or procedures

in the conventional procedural languages. The scriptable objects are further classifiable into those that can hold data and those that cannot. Their arrangement in the card-stack structure offers a friendly platform for both information acquisition and presentation. The pointing object, ordinarily a mouse, is an essential event-generation device to HyperCard as well as to the OS. A similarly important device, the keyboard, is omitted from the figure for the sake of perspicuity. The graphic object is auxiliary to HyperCard in ordinary use, but is of great value in psychological experiments. We explain next how the scriptable objects are organized into a HyperCard application file.

As the name suggests, the visual frontmost unit of HyperCard is a card, or more precisely, a rectangle window. On each card, one can place a number of button and field objects as its parts. Although buttons and fields are both data containable, the data in a button is not readily observable in contrast to the field data. Instead, a button name is visible (unless turned off), while a field name is basically concealed. Furthermore, a button can have an icon stored in the stack resource.

A group of cards of the same size comprise a stack together with a special layer called background. The background parts (buttons and fields) are visually observable underneath each card of the same stack. Put in reverse, a stack is a HyperCard application file consisted of a card layer and a background layer. On the card layer, one can exhibit, one at a time, the related information spatially organized into a window rectangle. The stack, however, is not a mere collection of the layers, but it is scriptable as explained below.

Insert Figures 2 and 3 about here

A script is a set of program statements to process messages by message handlers and functions. A handler begins and ends with a message name preceded by word 'on' and 'end', respectively. For instance, a button containing a handler shown in Figure 2 shifts its position along with the mouse until it is released. Terms 'the mouse' and 'the mouseLoc' are functions which respectively returns 'up' (or 'down') and the center coordinate of the mouse. Eventually, the handler mimics the basic behavior of a slider button, but needs refinements for the practical scaling purpose to be discussed in the next section.

Like any other object-oriented languages, one can send a message to another object to perform a task. Besides, HyperCard allows an uncaptured message to

travel along the predetermined message path until it reaches the HyperCard itself as depicted in Figure 3. Even after being captured, however, the message may be explicitly passed onto the path. Besides the mouse-event-driven messages, HyperCard also generates important system messages that are really useful in building an application. However, the gist of HypeScaling is comprehensible without such technical details.

As one can easily understand from the foregoing explanations, the efficiency of application software and the portability of objects depend largely on the organization of objects and message handlers along the message path. To add to it, the spatial arrangement and the appearance of buttons and fields will influence the usability. Although the usability is a crucial factor in reality, we presently make only a modest claim about it and leave room for improvement to interested readers. We now move to the discussion of the scaling methods and stimulus presentation.

The Scaling methods and stimulus presentation

Insert Figures 4 and 5 about here

---

1) Scaling methods and objects. Basically, a researcher presents a text or graphic stimulus (stimuli) on a HypeScaling card, and measures the responses of a subject by a set of sliders in a uni- or two-dimensional methods (see Figures 4 and 5). The researcher may choose either or both methods depending on the purpose of the study. In the present version, a slider is expressed by a card button with a linked icon, whereas an object for presenting a stimulus varies depending on its type as listed in Table 1.

Insert Table 1 about here

Concerning the uni-dimensional method, there are two transitional methods in addition to the basic ones shown in Figure 4: point→triple-values, and interval→triple-values. The transitional methods are expected to ensure the practical feasibility of the triple-value evaluation by reducing the initial complexity of responses. Although not recommended in terms of usability, the five methods may appear altogether on the same (card) window, if desired.

A slider moves along a scale axis in the uni-dimensional method or within

the scale area in the two-dimensional method. In either method, a scale object (an axis or an area) is expressed by a button with some ornamental graphics. As mentioned earlier, the data stored in a button is concealed to the viewer. This property is ideal for placing default values of an individual scale. Another advantage accruing from the use of a button over graphic painting is that the object determines the static boundaries for the slider. However, the multiple sliders of the uni-dimensional method impose dynamic boundaries on the partner slider(s). Obviously, the handler of Figure 2 is too simple to deal with the constraints. Since the refinements require elaborate considerations in order to reduce rigidity of control while maintaining portability, we explain them in the separate section.

2) Stimulus presentation. The experimenter should choose an appropriate object from the list shown in Table 1 for presenting a (visual) stimulus. If desired, then can appear simultaneously or sequentially on the same card.

Among the listed types, the paint graphic is not a HyperCard object unlike the others. Its sole merit is the ease of storage, being directly painted on a card. The merit of the HyperCard objects, on the other hand, stems from the fact that their positions can be changed according to the response. Therefore, one can easily design, for instance, a classificatory or a matching task.

A note seems in order about the differences among the parts button-field parts and graphic windows. Literally being the parts of a card or a background, the locations of buttons and fields are confined within the boundaries of a card window which is shared by the background. In contrast, a graphic window can be placed anywhere on the (virtual) screen, independently from the card window. Still, it is controllable by script. Color layer is in a sense, a pseudo-layer of a stack, since it is not scriptable unlike a card and a background, though it is script controllable like the external graphic windows. It is an internal layer in the sense that the graphics on the layer appears within the limits of the card window and beneath an overlapping button and field.

Auxiliary objects

To implement HyperScaling in a real experiment, one further need auxiliary objects for presenting instructions, stimulus names, scaling item labels, response feedback, filler graphics and so forth. Furthermore, one may desire to record responses in a (hidden) field or an external file. The hypertext-type

control of these objects should enhance the versatility of an application. Nonetheless, their discussions are beyond the scope of the present paper.

The Constrained Move of Sliders

A slider should move along with the dragged mouse. Straightforward as it may sound, it poses two technical problems. First, there is no single message or event that directly corresponds to a mouse dragging. The basic solution is shown in Figure 2 which, nevertheless, needs modification to handle multiple sliders on the same type. The second problem concerns the permissible range of the slider position. The mouse may go beyond the limits of the scale object (an axis or an area), but the slider itself should be confined within the scale boundaries that are static. Furthermore, sliders of the interval and the triple-value modes places the dynamic boundaries on each other. Let us illustrate below our programing strategy with a point and interval sliders on a horizontal axis.

Insert Figure 6 about here

These sliders differ in the position to control, since they are expressed as buttons, or more specifically in rectangles with respectively linked icons. While the scale value of the point slider corresponds to the center of the button, that of a lower-end (or upper-end) slider corresponds to its right (or left) edge as shown in Figure 6. Clearly these are the control loci for handling the two problems mentioned above. For the sake of brevity, they will be denoted by Pnt, Lo, and Up. Similarly, we denote the left and right ends of a scale axis by EndL and EndR, respectively. It should be noted that the scale value of a slider, V, is the relative distance of its control locus X from the left end of the axis to its width, i.e.,

$$(1) \qquad V = (EndL - X) / (EndR - EndL) \qquad X \in \{Lo, Pnt, Up\}$$

By adding a constant to or multiplying V, or applying both, one can easily convert V to the desired range of a scale, e.g., [0,10], and [-5, +5]. A slider in the two-dimensional method produces two such values along the horizontal and vertical axes.

Being confined in the static scale boundaries, Pnt, Lo and Up are all subject to the following rule:

$$(2) \qquad EndL \leq X \leq EndR.$$

The dynamic, mutual restrictions between Lo and Up in the interval mode, however, requires a slight change of the rule as:

(3) $\text{EndL} \leq \text{Lo} \leq \text{Up} \leq \text{EndR}$.

Further extension of the rule is necessary in order to incorporate the dynamic restrictions among Lo, Pnt and Up in the triple-value mode as

(4) $\text{EndL} \leq \text{Lo} \leq \text{Pnt} \leq \text{Up} \leq \text{EndR}$.

In the earlier version of HyperScaling, the dynamic constraints were imposed rigidly such that the Lo, for instance, was never allowed to move to the right of Up. The subject, then, was forced to shift Up first to increase the value of Lo. As one can easily imagine, such preparatory operations became more complicated in the triple-value mode. Demanding extra cognitive work, the rigidity greatly hampered the usability. The lessen led us to design a flexible control about the dynamic constraints.

The new control permits a temporary violation, but the constraints are soon to be restored by software. That is, the mutual restrictions among Lo, Up and Pnt in (3) and (4) might be temporarily violated while the mouse is being dragged, but to be restored immediately upon the mouse release. The restoration itself is, in a sense, a data-driven process similar to the slider placement by anchoring or by default. Eventually, they share a core module.

It must be clear by now that each slider type requires its own modules to efficiently implement the aforementioned ideas. The module design explained below is a prototype, and, hence, should be modified according to the experimental needs.

The module design

Placement of common modules. It would be hardly unrealistic to assume that researchers often attempt to measure responses to a stimulus or stimuli by multiple sliders of the same type. Were if HyperCard more advanced in object-orientedness, it allows a class definition for sliders that behave similarly. Instead, it provides the mechanisms of message sending and passing as explained earlier. To use explicit message sending, one simply designates the first slider as the object representing the group to which the remaining slider explicitly sends a relevant message. Alternatively, one may place the common modules in an object on the message path, i.e., the card, the background or the stack. More advanced strategy is to create a separate HyperCard stack, say

"sliders", and place it before the "Home" stack in the message path by a single command line:

```
start using stack "sliders"
```

The merit of this library feature is appreciable. The script in the stack can be used in any other experimental stack as long as the above command is declared. Although we prefer message passing to sending, the following illustrations are pertinent to both.

Composition of the common modules. The shift of a slider begins with a "mouse down" event on the object and ends when the mouse is released. During the process, the scale value of the slider needs to be computed according equation (1) while observing restrictions (2), (3) or (4). Furthermore, a timer must be set, if response time or any temporal information is desired. It is suggested, for the subsequent analysis, to export these records to an external file with the slider's name. These and other related considerations led us to attach a simple message handler to each slider that issues the slider specific message with the appropriate parameters (see Figure 7). The latter message will be captured by the corresponding handler. The main components of the parameters are the name of the slider and its coordinates.

Insert Figures 7 and 8 about here

The `response measurement` segment is actually an extended repeat loop for monitoring the mouse dragging shown in Figure 2. If appropriate, the ongoing scale value can be displayed during the process. When the loop ends upon the release of the mouse, a temporary violation of the constraints (3) and (4) will be recovered by software. It must be clear that such recovery is not necessary for the two-dimensional method at all and for the single-value mode of the uni-dimensional method. This is one of the principal reasons to build slider specific modules.

The software recovery is a data-driven process, involving the inverse function of (1) with respect to X such that

(5) $X=V*(EndR-EndL)+EndL \qquad X\in\{Lo,Pnt,Up\}$

where V=Up (or Lo) and X = Lo (or Up) if (3) is violated in the interval mode, i.e., Lo > Up upon the mouse release. In the triple-value method, the violation must be checked against the two partner sliders. The lower-end slider violates (4), if

$$Pnt < Lo < Up \text{ or } Pnt \leq Up < Lo.$$

The former requires the single recovery, i.e., X=Pnt, while the latter requires the double recoveries, i.e., X=Pnt, Up. Similar recoveries will be made for the violations by the other sliders.

Function (5) plays an important role also in placing the remaining slider(s) of the transitional modes. For instance, if the point slider is to appear as the mathematical or geometric mean of the interval ends, it should contain the following function in a special handler, say, called "`showUp`" (see Figure 8). The function derives the X-coordinate from the values of Lo and Up. On may regard this as an instantaneous anchoring. Noninstantaneous anchoring, in contrast, makes use of the slider placement by default values stored in a value-container object such as a scale bar or a scale area in the uni- and two-dimensional methods (see Figures 4 and 5), respectively. Visual adjacency to the affected object, i.e., the slider, helps information management. An alternative choice is a more adjacent object, the slider itself, which incurs, however, some inconvenience in managing more than one default values in the multiple-value and the transitional modes.

It goes without saying that the dynamic use of default values will greatly increase the practical plausibility of contingent experiments.

Discussion

Scaling responses by sliders is just the first step of HyperScaling which is intended to encompass a wide range of linked text and audio-graphics information. We introduce below two extended use of HyperScaling to show its empirical value in the psychological research. Both of them involve placement of graphic stimulus by a pointing device.

Semantic differential has been widely used in pyschology and many other related fields since the seminal work of Osgood, Suci and Tannenbaum (1957). Although it's merit is undeniable, it is our conviction that the affective meaning of a concept or an entity is not always readily expressible in words particularly in *kansei* information processing that has recently caught heavy interdisciplinary attention in Japan (e.g., Tsuji, 1995). In short, *kansei* constitute the base of our intellectual faculties at the top of which rationality is situated. Thus, *kansei* should not be considered irrational. Rather, it is complimentary to rationality, being lenient to deviations from completeness, consistencies, exactness, rigor and other properties of rationality (Matsuda & Namatame, 1995). These considerations led us to develop semantic-differential (SD) type measurement via matching and grouping.

Designers of consumer products often employ object-matching techiniques in stages where comprehension by *kansei* is more suitable than by rationality. In designing a new video camera, for instance, they ask a monitor to match the prototype model with one of the (graphic) images of various activity scenes on the intuitive base. The process has produced many useful results. These field practices inspired us to arrange a matching task between jewelry pictures and brand-logos (Matsuda & Namatame, 1995). Previously measured SD scores of jewelry pictures were later used as anchors in measuring the impressions of the logos. The post-experimental interviews indicated that the intuitive preprocessing helped subjects form impressions about the several stimuli because verbal interpretations and numerical expressions were not forced in the incubative stage. While this matching required perceived equivalence of the objects in impressions, the following grouping-and-rating task involves similarity judgments of objects and judgments by representativeness.

People tend to deal with objects around them in categories defined by

academic, social or personal standards. In their seminal work, Rosch and Mervis (1975) demonstrated that the prototypical members of a category would have more attributes in common than do those members judged non-prototypical, employing both natural and artificial category/objects. Though not explicitly excluded from their study, affectionate attributes were not of their concern. If their findings are truely general, a prototypical member should exhibit more similarities, for instance, in SD-ratings with other members of the same category than with those of the other categories.

People also tend to make judgments by representativeness (Kahneman, Tversky & Slovic, 1982) in that a judgement about an object is likely to be influenced by the degree to which it represents a category or to which it resembles the typical member of the category. Then it is natural to expect that the grouping and the identification of a representative member that preceed the SD-ratings should enhance the dominance of the prototype mentioned above. The effect would be more enhanced, if the ratings of a representative member are provided as a starting value of the other instances of the same category, i.e., internal anchoring in our terminology. An experiment is in progress to test this idea with an extended version of HyperScaling.

The foregoing explanations might have left an impression that HyperScaling is applicable chiefly to the *kansei* judgments and SD-type ratings. On the contrary, the software is also suitable for the rational decision making such as probability judgments and pricing.

When we were about to finish drafting the paper, we learned the release of new visual programming software called Oracle Media Object (Oracle, Co. Ltd.), OMO for short. To our best knowledge, it is highly compatible with HyperCard with respect to the stack-object structure, the programming language and the message hierarchy. One of the noticable differences is the property of graphics. In OMO, graphics may be treated as scriptable objects just like buttons and fields. This feature enhances efficiency in both script management and visual interface. Moreover, the new software is intended to be cross-OS. All these facts ensure the general applicability of our methodology. In fact, we are presently converting our HyperCard stacks into OMO stacks which will soon become available upon request.

References

Booch, G. (1991). *Object oriented design with applications*. Redwood City, CA: Benjamin/Cummings.

Hesketh, B., Pryor, R., Gleitzman, M., & Hesketh, T. (1988). Practical applications and psychometric evaluation of a computerized fuzzy graphic rating scale. In T. Zétényi (Ed.) *Fuzzy sets in psychology.* Amsterdam: Elsvier Science Publishers.

Kahneman, D. (1992). Refernce points, anchors, norms, and mixed feelings. *Organizationla Behavior and Human Decision Processes*, 51, 296-312.

Kahneman, D., Slovic, P., & Tversky, A. (1982). *Judgment under uncertianty: Heuristics and biases*. New York: Cambrdige University Press.

Matsuda, N., & Namatame, M. (1995). Interactive measurement of the hierarchically related consumer's images. *Behaviormetrika* (in press).

Matsuda, N., & Shinoda, N. (1995). The point and interval measurement of automotive risks in the top-down procedure. The paper presented at the annual meeting of the Society for Risk Analysis. Honolulu.

Matsuda, N., Shinoda, N., & Takemura, K. (1995a). The effects of the top-down and the bottom-up procedures on the perception of automotive risks: A triple-value approach. (submitted)

Nielsen, J. (1993). Hypertext and hypermedia. Cambridge, MA: Academic Press.

Osgood, C.E., Suci, G.J., & Tannenbaum, P.H. (1957). The measurement of meaning. Chicago, IL: University of Illinois Press.

Rosa, E.A., & Humphreys, P.C. (1988). A decomposition approach to measuring human error probabilities in nuclear power plants: A case example of the SLIM-MAUD methodology. In P. Humphreys, O. Larchev, O., Vari, A., and J. Vecs (Eds.) *Strategic decision support: Frames and case studies.* IIASA, Amsterdam: North Holland.

Rosch, E., Mervis, C.B. (1975). Family resemblance: Studies in the internal structure of categories. *Cognitive Psychology*, 7, 573-605.

Tsuji, S. (1995). (Ed.) *Informatics and psychological research on Kansei information processing.* Report No. 04236107. Special focused research project, the Ministry of Education, Science and Culture.

Tversky, A. (1994). Support theory: A nonextensional representation of subjective probability. *Psychological Review*, **101**, 547-567.

Table 1
Basic set of objects by stimulus type

stimulus	object
Text	field, button (name)
PAINT graphic	card or background layer
PICT graphic	picture window, button (icon)
	color layer
QuickTimeMovie	movie window

```
on mouseDown
   repeat until the mouse is up
     set the loc of me to the mouseLoc
   end repeat
end mouseUp
```

Figure 2. A script example

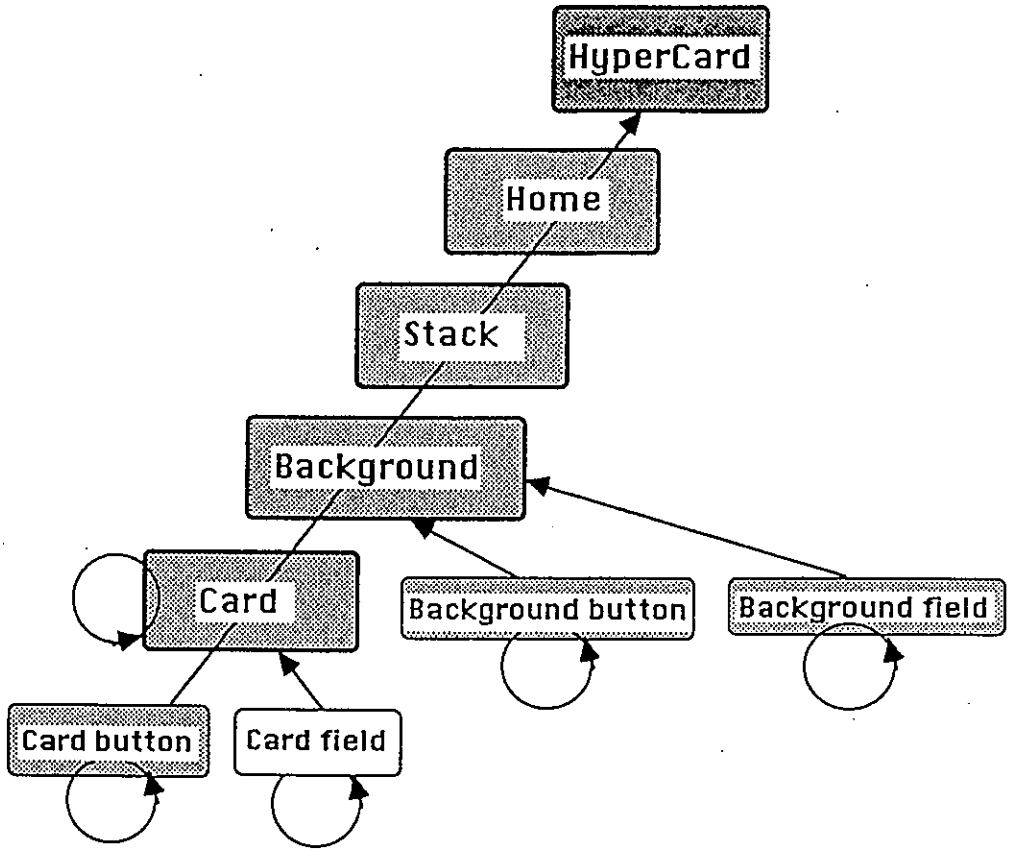

Figure 3. The message path

Notes: 1) The topmost "HyperCard" is the software, while "Home" is a pecially designated HyperCard strack.
2) The objects with an arc may directly capture the mouse-related messages.

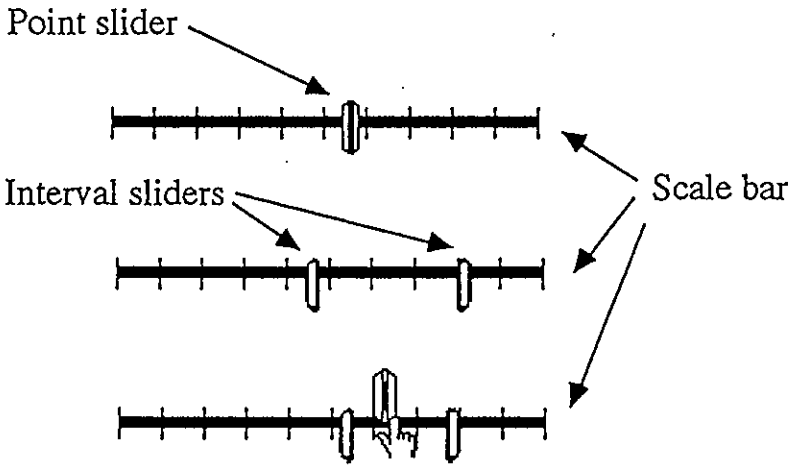

Figure 4. Three basic methods of the unidimensional mode

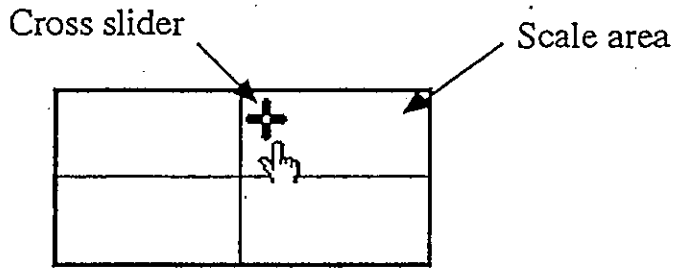

Figure 5. The two-dimensional mode

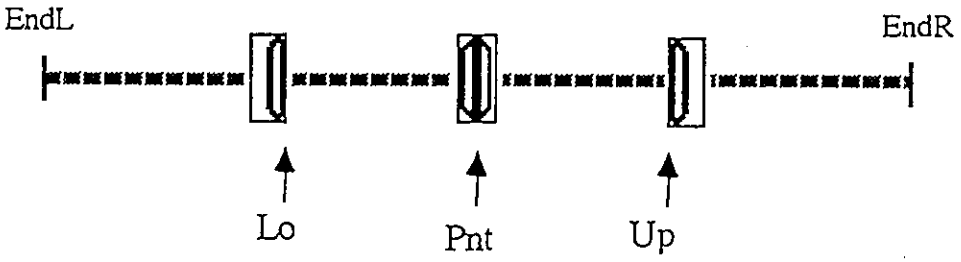

Figure 6. Positions to be controlled.

a) The script attached to each slider:

```
on mouseDown
    sliderSpecificMessage with parameters
end mouseDown
```

b) The script stored in the background

```
on sliderSpecificMessage parameters
    timer setting
    response measurement
    recovery of a temporary violation
    transition to triple-value measurement, if required
    exporting response records
end sliderSpecificMessage
```

Figure 7. Recording script

```
on showUp
   put mean(Lo,Up) into X; or put sqrt(Lo,Up) into X
   resetPnt X
   show me
end showUp
```

Figure 8. ShowUp script