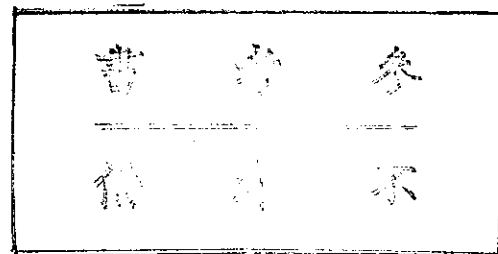


TR-87-012

AN EXTENDED VISUAL PROGRAMMING  
SYNTHESIZER FOR COMPUTER  
AIDED INSTRUCTION APPLICATIONS



中研院資訊所圖書室



0072

**TR-87-012**

**An Extended Visual Programming  
Synthesizer for Computer  
Aided Instruction Applications**

**K.Y.Cheng, M.S.Hwu, and C.C.Hsu**

**Institute of Information Science  
Academia Sinica  
Taipei, Taiwan, R.O.C.**

## ABSTRACT

An extension work to our previous visual programming synthesizer for computer-aided-instruction (CAI) application is presented. The extended visual programming synthesizer is to increase the representational power by including the Student-Problem curve analysis for the development of smart CAI programs.

This work is supported by the National Science Foundation of R. O. C. under granted No. NSC 76-0408-E002-12 December, 1986.

## 1. INTRODUCTION

In our previous work[3], we have tried to design a form-oriented and visual-directed system for CAI courseware developing purpose, in which a visual programming synthesizer called VIPS was developed so that end users with little or no knowledge of computer programming can use it to develop their own applications(coursewares). It is known that all form-oriented approaches are application dependable, i.e, different applications require different form representations and structures. For instances, in office information systems application, QBE/OBE uses a two-dimensional one-level skeleton table[16,17] and FORMAL uses a two-dimensional representation of hierarchical data[10,11,12], and, in CAI courseware developing systems application, form-driven approaches were widely used, where forms are represented and structured in linear strings[4,6,8,9]. The purpose of proposing an extended VIPS is to include with more representational power to upgrade the level of its application.

The nature of the original VIPS was set to the application level of courseware content creation and lesson definition. With this level of applications, the input, formatting, and modification of text, graphics, audio, or any other information can be created and displayed, also each lesson's control structure can be defined. The limitation of these applications is its deficiency in lack of courseware analysis and progress evaluation, because in this level of applications, there is no recording and analysis of data generated during student's learning process. In order to enhance the capabilities of VIPS in this respect, we adopt the so called Student-Problem (S-P) curve analysis[14].

The Student-Problem curve analysis requires to record a chart concerning

the attributes of problem and each student scores( right or wrong) to each problem as a (0, 1)-matrix, which contains the information of the characteristics of the courseware versus the student learning process. For example, the rearrangement of an obtained (0, 1)-matrix according to the descending order of student scores rowwise and the descending order of correctly answered problems columnwise forms two boundary curves, the S-curve and the P-curve, whose relative shapes and local properties contain a lot of information such as the distribution of problem being answered, a certain student reaction to certain kind of problems; and so forth. Therefore, in this paper, we propose an extended VIPS which is capable of recording and reordering each of the obtained (0, 1)-matrix, extracting information from them, and printing the final analysis results to inform the teacher the effectiveness of the courseware as well as student learning behavior as a reference for further improvement or consolation of the developed courseware.

## 2. THE SYSTEM ORGANIZATION

The extended VIPS consists of four major parts: a Visual Interface, a V-Form Interface Language, an Internal Structure, and a V-Form Interpreter. Fig.1 shows the system organization.

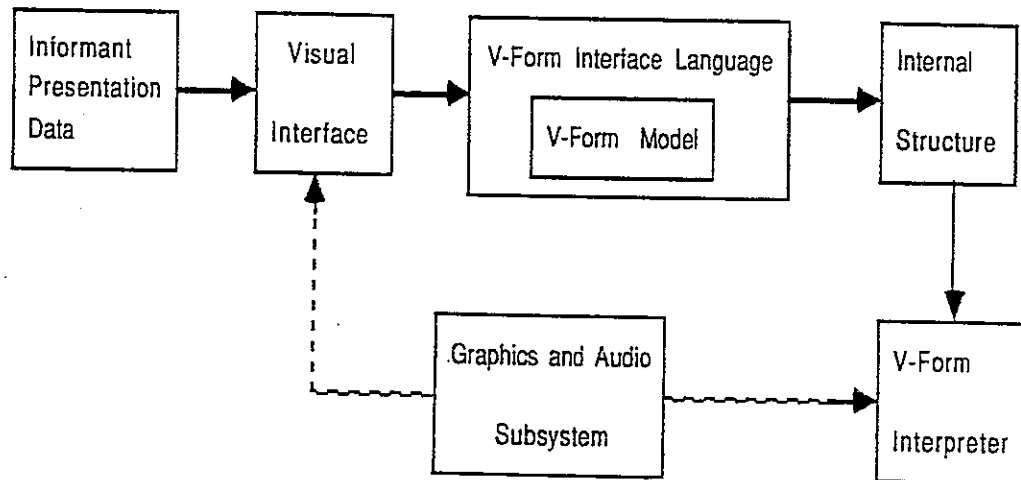


Fig. 1. The system organization of the extended VIPS.

The Visual Interface provides an environment for users to communicate with the system through a medium called the V-Forms. The Informant Presentation Data of V-Forms consists of text, static graphics (line drawing and bit-map), dynamic graphics (animation), and voices. The V-Form Interface Language is used to define and manipulate V-Forms under a V-Form model, in which each V-Form consists of a V-Form type to specify both the V-Form logical structure (scheme) and visual structure (template) and a V-Form instance to be filled with the presentation data[3], V-Forms thus constructed form the external structure of the V-Form model. V-Forms in the external structure are automatically converted into an internal structure to hold the information of the specified V-Forms and the control flow of the defined courseware (application program). The V-Form Interpreter interpretes the internal structure to achieve all functions of the application program.

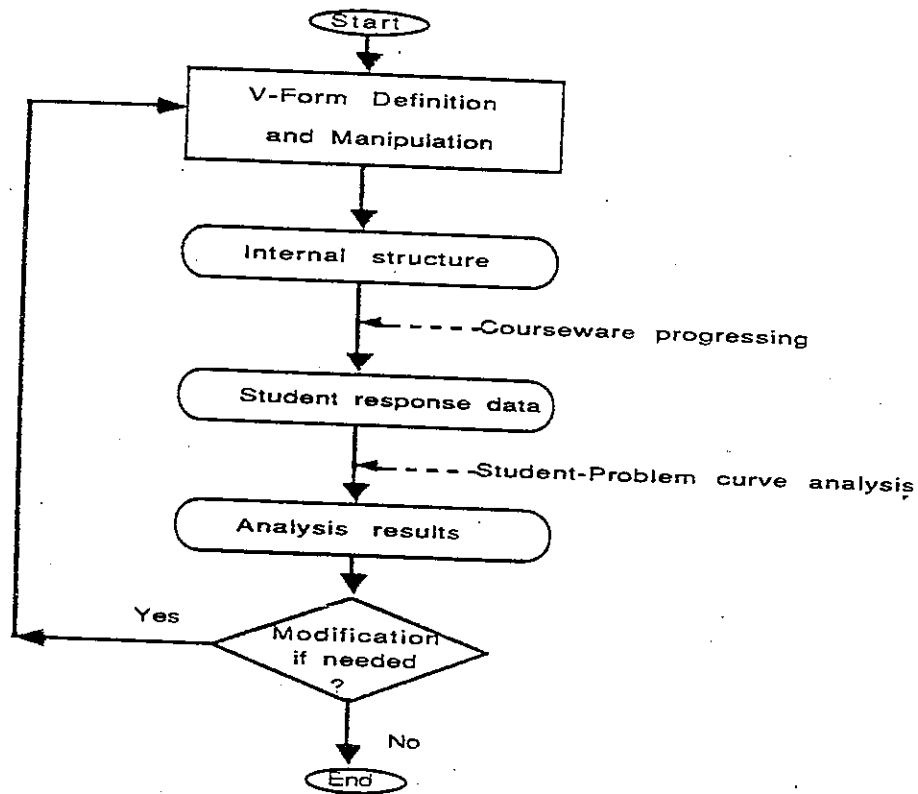


Fig. 2. The application program construction flow .

Fig. 2 shows the construction flow of an application program. A courseware designer first uses the system provided V-Form interface language to define and manipulate all V-Forms which are then self-synthesized as an internal structure to become executable by the V-Form interpreter. After progressing the courseware by students, the student response data are collected into a table (which contains a  $(0, 1)$ -matrix). The teacher then selects from a list of all possible analysis results those items of what he wants to know, for example, the distribution of types of problems being correctly answered. The analysis results are obtained from various interpretations of the formed Student-Problem curve, the print out may suggest the courseware designer to modify the material of the lesson if the students learning performance is unsatisfactory.

### 3. THE INTERACTIVE V-FORM PROGRAMMING

The interactive V-Form programming uses forms as media for programming and employs multiple windowing and interactive facilities used in most icon systems[1,2,5,7,13,15]. Fig. 3 shows a V-Form programming screen, where the top area shows a command type menu, each command type when selected can pop-up a new set of operation commands underneath it, the rest screen area is a region for programming in a manner of what-you-sketch-is-what-you-get.

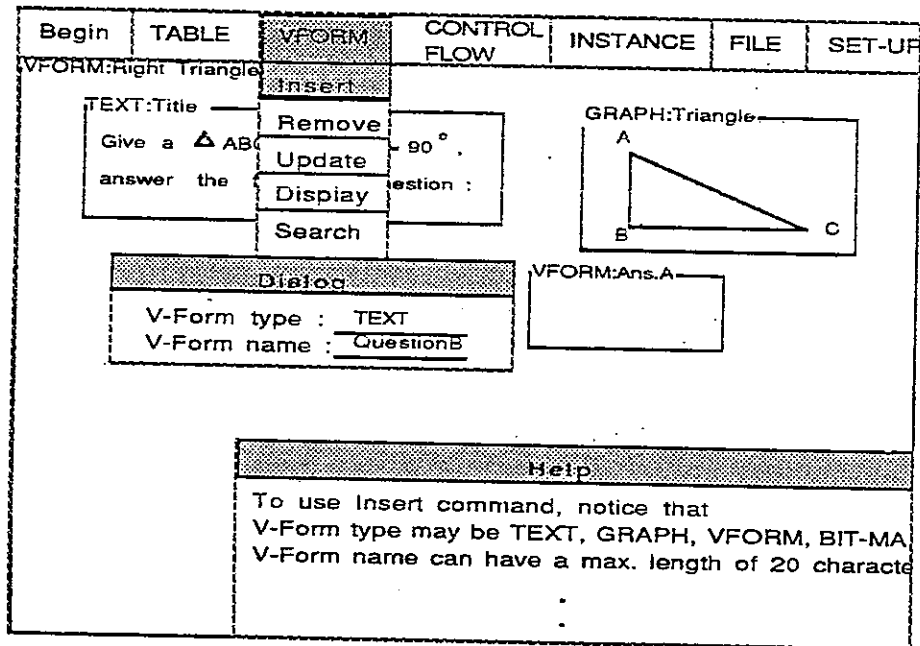


Fig. 3. The VFORM command type operations.



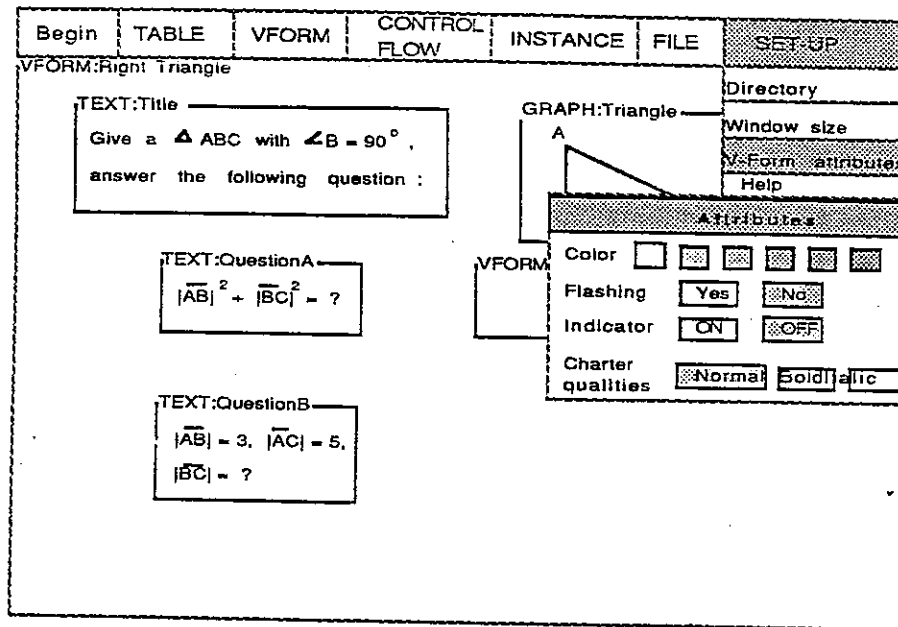


Fig. 4. The SET-UP command type operations and the V-Form attributes.

Fig. 3 and Fig. 4 illustrate some typical steps for the creation of a V-Form and its reaction in screen displaying. The selection of VFORM command type pops up five operation commands as shown in Fig. 3. Then the selection of "Insert" operation command displays a dialog window and possibly a help window (if it was specified in the set-up as an "on" status). The user then fills in a V-Form type and a V-Form name into the dialog window, the TEXT type and QuestionB name in this example. After that, both windows disappear to allow the user to locate a TEXT window and to fill in the content of the V-Form QuestionB in the working area. The selection of SET-UP command type pops up four operation commands as shown in Fig. 4. Then the selection of the "V-Form attributes" operation displays an attribute window to overlay the previous screen. The selection of the items listed in the attribute window sets up the display attributes of the TEXT:QuestionB. Notice that, all window handlings are under the

"Window size" operation of the SET-UP command type.

Begin	TABLE	VFORM	CONTROL FLOW	INSTANCE	FILE	SET-UP
	Define UDT	Insert	Create	Group	Open	Directory
	Define RAT	Remove	Extend	Degroup	Close	Window size
	Edit ECT	Update	Link		New	V-Form attributes
		Display	Remove		Save	Help
		Search			Save As	
					Print	
					Exit	
					Quit	

Fig. 5. Summary of operation commands.

Fig. 5 shows a summary of operation commands in each command type. The "Begin" operation command allows a user to define a system name and then performs initialization for courseware creation. The "Search" operation command of VFORM command type can be used as a page-turner which allows the user to search for and display out each of the turned V-Form pages (a page contains all V-Forms within a programming window). The construction of V-Forms into pages and their structures is through "CONTROL FLOW" command type, the operation command "Create" is used to create an independent new V-Form page, "Extend" is used to extend an existed V-Form (with VFORM type) and create a new V-Form page. "Link" builds a physical link between two existed V-Forms, and "Remove" is an inverse operation of "Link".

The importance of including the "TABLE" command type in the menu is to

record data generated during the execution of the developed application program. There are three operation commands in the "TABLE" command type, a User Data Table (UDT) to define user's information of the courseware (Fig. 6(a)), a Record Attribute Table (RAT) to specify each (problem) attribute name and its classification, for example, problem#4 belongs to an abstraction type problem (Fig. 6(b)), and an Embedded Control Table (ECT) to specify the embedded control section of a V-Form page (Fig. 6(c)).

File name of the user's records: <u>user.dat</u>	
User data specification	
< Prompt string >	< Input type >
User name :	CHAR(20)
ID number :	CHAR(7)
.	.
.	.
.	.

(a)

Attribute name	Category
P1	Definition
P2	Computation
P3	Deduction
P4	Abstraction
.	.
.	.
.	.

(b)

VFORM:Right Triangle

TEXT:Title

Give a  $\triangle ABC$  with  $\angle B = 90^\circ$ ,  
answer the following questions :

TEXT:QuestionA

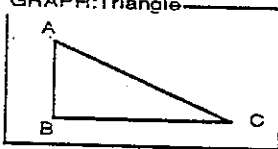
$|\overline{AB}|^2 + |\overline{BC}|^2 = ?$

TEXT:QuestionB

$|\overline{AB}| = 3, |\overline{AC}| = 5,$   
 $|\overline{BC}| = ?$

VFORM:New  
New

GRAPH:Triangle



VFORM:Ans.A

VFORM:Ans.B

VFORM:Exit  
Exit

(c)

ECT					
	Auto	Ans.A	Ans.B	New	Exit
Input type		CHAR(5)	NUM(1)		
Matched conditions		EQ "AC"²	EQ 4		
Execution order		1	2		
Match control actions	delay 3 display QuestionA display Ans.A	set v1=1 display QuestionB display Ans.B	set v2=1 display New display Exit		
Eise actions		set v1=0 display QuestionB display Ans.B	set v2=0 display New display Exit		
Records		P1 : v1	P2 : v2		

Input type

Match condition

Execution order

Match control actions

Eise actions

Records

(d)

Fig. 6. The TABLE operation commands: (a) the User Data Table (UDT), (b) the Record Attribute Table (RAT), (c) an illustrated V-Form page VFORM:Right Triangle and (d) its Embedded Control Table (ECT).

After the construction of a V-Form page, and the selection of the "EDIT ECT" of the TABLE command type a skeleton table is automatically generated to allow the user (courseware designer) to fill in the appropriate data in the corresponding items. The menu in the bottom area of ECT window is used to choose the item to be filled in. The system then provides the capability of type checking on the execution order item and syntax checking on all other items during the editing of the embedded control table.

#### 4. THE EXECUTION OF V-FORMS

As previously described, when an application program (in terms of V-Forms) has been completely constructed by a courseware developer, it is automatically converted into an internal structure to hold the courseware program control flow and the information of the specified V-Forms. The application program thus constructed can be executed by the interpretation of its internal structure. The internal structure is basically a graph which consists of five different kinds of nodes and their interconnections. These five kinds of nodes are System Node(SN), Form Control Table(FCT), Form Control Node(FCN), Action Node(AN), and Mode Node(MN), respectively.

An application program has only one SN node to keep the system information, such as its system name, pointers that connect the system to an User Data Table, a Record Attribute Table, and to the rest of the system. The tail terminal of the pointer from SN node to the rest of the system is a FCT node which keeps the information of a V-Form page such as page margin, pointers that connect to the control actions in its ECT table (the "Auto" field in Fig. 6(d)) and to a linked list of a sequence of FCN nodes. A FCN node keeps the information of a V-Form, such as the V-Form name, its window size and location, input type, input length, execution order, a control pointer that connects to an AN node, a content pointer to a linked list of a sequence of MN nodes and a pointer to another FCT node. An AN node holds three pointers, a condition pointer, an action pointer, and an else pointer, respectively, each points to a linked list of a sequence of MN nodes with the information of condition, action, and else as shown in Fig. 6(d). A MN node records the information in two parts, the first part is the displaying attributes of a specified fragmented content (a content may need several MN nodes to keep within it), the second part is the content portion which may be

part of the content of a V-Form or the condition, action, else, and record specified in ECT table.

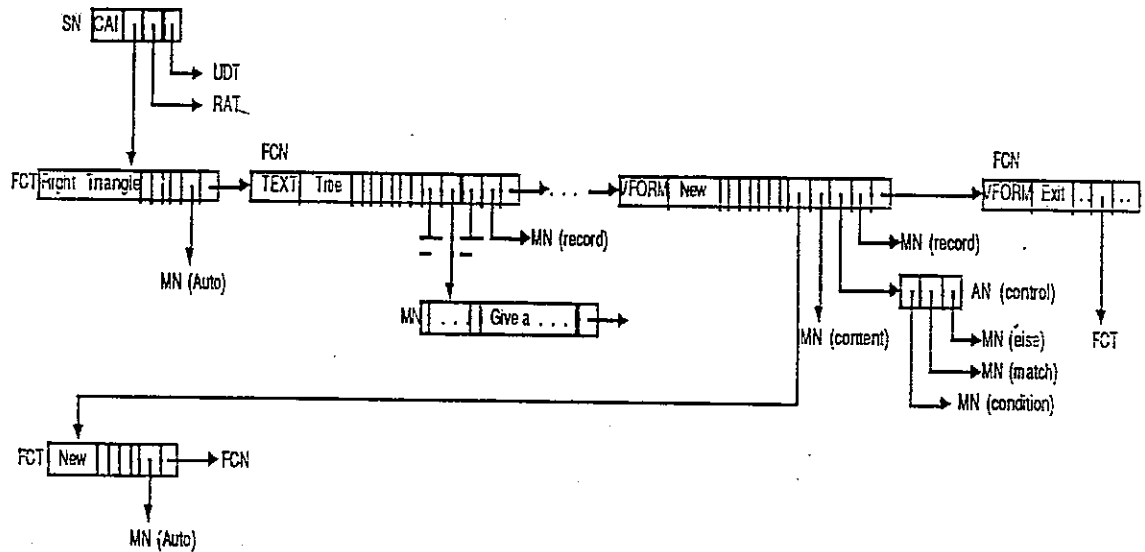


Fig. 7. The internal structure of VFORM:Right Triangle.

The execution of V-Forms is through the interpretation of the internal structure by an interpreter. Fig. 8 shows the internal structure synthesized by the external structure of the application shown in Fig. 6(c). We use it as an example to explain the interpretation process. The interpreter starts from the SN node of an application program specified by the system name (CAI in this example) and then allows each user (student) to fill in the information specified in the User Data Table (for example, student name and ID number). After that, the system goes to the first FCT node to start the interpretation of the internal structure of the first V-Form page (VFORM:Right triangle in Fig. 6(c)) which is chained as shown in Fig. 7. The window margin of FCT defines the window of VFORM:Right Triangle and the Auto actions defined in Fig. 6(d) will be executed after all V-Forms in the first page have been displayed. In each FCN node, the window size

and location define a V-Form window, the content pointer incurs the content of the V-Form to be displayed. Now, the V-Form TEXT:Title and GRAPH:Triangle are displayed while the rest V-Forms are not displayed because they were set at "OFF" status (Fig. 4), and after delay 3 time units (as specified in Auto) the TEXT:QuestionA and the VFORM:Ans.A are displayed. At this time, the system finds a FCN node with VFORM type to allow an user giving his answers according to the execution order, if the question is correctly answered (i.e, satisfy the matched condition,  $|ACI|^2$  in this example) then the system performs the match control actions else performs the else actions according to the information of the AN node pointed by the control pointer of the FCN node. The match control actions set a value  $v1=1$  while the else actions set  $v1=0$  for Ans.A and record  $v1$  value in Record field of the ECT table as P1:v1 through the record pointer of the FCN, then both actions display the TEXT:QuestionB and the VFORM:Ans.B and repeat the process (for the student to continue his answer on QuestionB). Notice that the input type of "CHAR(m)" and "NUM(n)" is for students to fill in the blanks while that of "\*" is for students to perform the selection as specified by the courseware developer (New and Exit in this example). The matched conditions of New and Exit are both empty, when either one is selected the matched condition is automatically satisfied and the system goes to the specified FCT. However, if there is a non-null FCT pointer in a FCN node and when the match conditions are satisfied during the interpretation of the FCN node then system branches to the pointed FCT. Finally, a (0, 1)-matrix can be constructed from the Record field of ECT table after giving a test to a group of students.



## 5. THE S-P CURVE ANALYSIS

After giving a test to  $m$  students with  $n$  problems a record of  $P_i:v_j$  for each student is formed (according to the Record fields of ECT tables specified by the courseware developer), where  $P_i$  denotes the  $i$ -th problem category and  $v_j$  is a value to denote the correctness of his answer to  $j$ -th problem, 1 for right answer and 0 for wrong answer, and so a  $m \times n$  (0, 1)-matrix is obtained. Fig. 8 shows a chart for 31 problems by 38 students. This chart can be rearranged according to the descending order of the student with high score in rows and the descending order of the problem with high correctly answered in columns, then a S-P chart is obtained in which two boundary curves, the S-curve and the P-curve, are formed as shown in Fig. 9. This S-P chart is useful for identifying unusual response patterns on the tests (in addition to the test score). For example, when both curves sharply tend to upper-left corner may indicate that the problems are very difficult for most students (Fig. 10(a)), on the other hand, when both curves sharply tend to lower-right corner may indicate that the problems are too easy (Fig. 10(b)), and so normally both curves should alternate between the diagonal line drawn from upper-right to lower-left corners (Fig. 10(c)). Also, new information may be gained by adding some measures to categorize the S-P chart to help for further diagnosis on each student's understanding of specific skills. For example, a large caution index as shown in Fig. 9 indicates a student's unusual response in the phenomena that he can answer a few difficult problems but can't answer considerably many easy problems (#10 student with two dots in caution signal). Similarly, a categorized S-P chart may be formed if  $n$  problems are further divided into  $k$  categories, i.e.,  $P_i$ ,  $i = 1, \dots, k$ .

The results of S-P curve analysis can be expressed as a set of analysis

service items for a teacher to select what he wants to know about the diagnosis of the courseware and the evaluation of the learning process. The analysis service items provided in the extended VIPS are: S-P chart, S-Test score, S-Caution signal, S-Caution index, Number of students getting correct answer for each problem, Passing rate of each problem, P-Caution signal, P-Caution index, Average passing rate, and Disparity coefficient (see in Fig. 9). The selection of each of such items incurs a procedure to perform the computation of the related data and print the results in natural language output forms (deduced from the logical expression in Horn clause). For example, the natural language output may be of the form: "The lesson is too easy (or too difficult). The reason is because the disparity coefficient is greater than 0.6 (or less than -0.6)".

## 6. CONCLUSION

In this paper, an extended VIPS from our previous work is proposed with the purpose to show that smart CAI coursewares can be developed under a form-oriented and visual-directed programming environment. This is achieved by recording and analyzing data generated during the execution of the courseware. The extended VIPS provides a table that can record the generated data into as a Student-Problem chart. And, based on the Student-Problem chart, a Student-Problem curve is obtained such that from it useful information can be extracted and analyzed to behave a smart courseware.

\*\*\* ORIGINAL DATA \*\*\*

PROBLEM NUMBER

STUDENT NUMBER	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38					
C.A	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1				
1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1			
2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
3	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
4	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
6	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
10	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
11	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
12	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
13	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
15	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
18	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
19	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
20	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
21	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
22	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
24	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
26	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
27	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
28	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
29	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
30	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
32	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
34	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
35	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
36	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
37	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
38	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fig. 8. An example of a matrix for 38 students on 31 problems. (Courtesy of Takahiro SATO of C&C Systems Research Laboratories NEC Corporation)

XXX S-P TABLE XXX

Test score  
(Correct answers)

PROBLEM NUMBER  
easy ← → difficult

1 1 1 111 2211211222222223

STUDENT NUMBER	27	12	33	45	56	69	94	98	170	291	783	498	561	CA	%	C.S
16	11111111111111111111111111111111	31	100	0.00												
24	11111111111111111111111111111010	29	94	0.04												
36	1111111111111111111111111110111011101	28	90	0.41												
27	111111111111111111111111111111101001	27	87	0.38												
38	1111111111111111111111111110111110110000	25	81	0.22												
12	111111111111111111111111111001100110	25	81	0.22												
12	11111111111111111111101010110111110110	25	81	0.73												
11	1111111111111111111110110111110110110110	24	77	0.27												
2	1111111111111111111110111101110011100101	24	77	0.44												
32	111111111111111111111111111000100000	23	74	0.02												
19	111111111111111111111111111000000000	22	71	0.00												
23	111111111111111111111111111000100000	22	71	0.25												
3	111011110111111111111111111100000000	22	71	0.30												
10	1111101111100100100101111110111	22	71	1.13												
34	1111111111011111111111101111001000000	21	68	0.15												
39	1111111111101111111111101000000100	21	68	0.17												
29	111110110011111110110111110001100	21	68	0.63												
25	1111111111111111111111101000000000	20	65	0.01												
22	111111111111111111111011011100000000	20	65	0.04												
20	111111111111111111111001111000000000	20	65	0.05												
5	1111111111111111111110010000100010	20	65	0.30												
21	011101111111111111111010000001010	20	65	0.43												
15	1111111011010110111011101000100000	18	58	0.29												
6	111101001111101010101101110010000	18	58	0.56												
26	1111111110001101100000110011001	13	59	0.73												
9	11111111011101111110010000000000	17	55	0.11												
4	10011111111111011110101000000000	17	55	0.18												
1	100111111101111111001000110000000	16	52	0.26												
37	111011001000010101100010011000	13	42	0.73												
13	11111101110001001000100000000000	11	35	0.10												

high ↑  
↓ low

S-curve  
P-curve

(Correct answers)

CA 2722222222222222222222222211111  
9829267775555544322100332109870 ← Number of students getting correct

% 9999999992389295777766444333222  
7222330003223306733077330720730 ← Passing rate of each problem

C.P ..... ← Caution signal

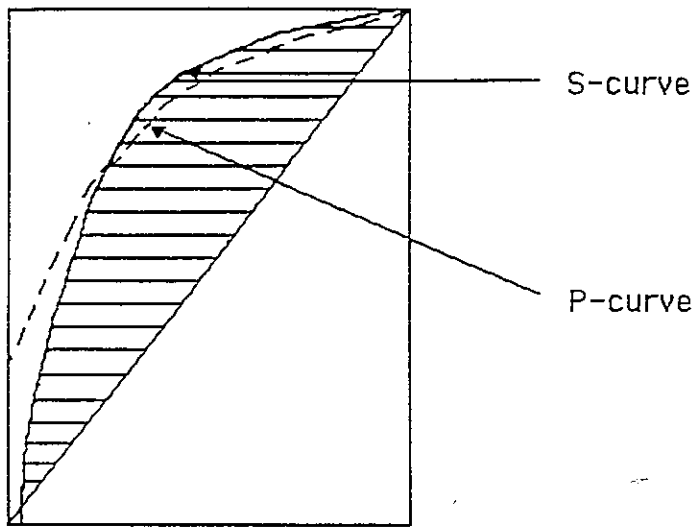
Caution index 00000100000000000000000000000000  
.....  
5445700301225550626623460256344  
6320513725145380961465264993941

TOTAL MEAN 62.2% ← Average passing rate

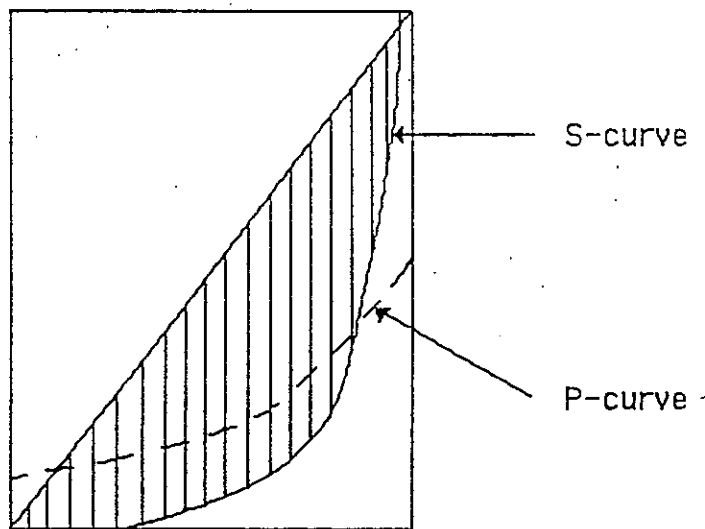
STUDENT	MEAN	SD	ATT. MEAN
27	21.3	4.4	0.31
PROBLEM	20.5	7.4	0.49

D=0.40 ← Disparity coefficient

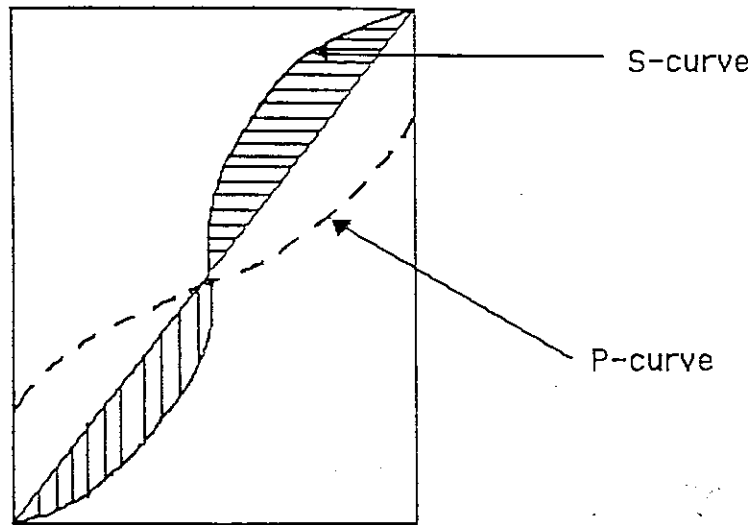
Fig.9. The S-P chart and the S-P curve analysis. (Courtesy of Takahiro SATO of C&C Systems Research Laboratories NEC Corporation)



(a) very difficult problems  
negative disparity coefficient



(b) very easy problems  
positive disparity coefficient



(c) norm reference test  
 normal distribution  
 near zero disparity coefficient

Fig. 10. Examples of S-P Charts.

### REFERENCES

- [1] Cattaneo, G., Guercio, A., Levialedi, S. and Tortora, G., "IconLisp: An Example of a Visual Programming Language", Second IEEE Computer Society Workshop on Visual Language, (June 1986) pp.22-25.
- [2] Chang, S.K., Shi, Q.Y., and Yan, C.W., "Iconic Indexing by 2D Strings", Second IEEE Computer Society Workshop on Visual Language, (June 1986) pp.12-21.
- [3] Cheng, K. Y., Hsu, C. C., Lin, I. P., Lu, M. C., and Hwu, M. S., "VIPS: A Visual Programming Synthesizer," Second IEEE Computer Society Workshop on Visual Language, (June 1986) pp.92-98.

- [4] Dowsey, M.W., "Easy author-entry systems: A review and prototype", *Int. J. Man-Machine Studies* 4, (1974) pp.401-419.
- [5] Glinert, E.p. and Tanimoto, S.L., "Pict: An Interactive Graphical Programming Environment", *IEEE COMPUTER*, Vol. 17, No. 11, (Nov. 1984) pp. 7-25.
- [6] Kearsley, G., et al., "Authoring Systems in Computer Based Education," *Comm. of the ACM*, Vol. 25, No. 7, (1982) pp. 429-437.
- [7] Musen, M.A., Fagan, L.M., and Shortliffe, E.H., "Graphical Specification of Procedural Knowledge for an Expert System", *Second IEEE Computer Society Workshop on Visual Language*, (June 1986) pp.167-178.
- [8] Olivier, W.P., "CAN-8: A complete instructional system", In *Proceeding of the Association for Canadian Instructional Technology Conference*, Vancouver, Canada, (Dec. 1980).
- [9] Schulz, R.E., "Lesson MONIFORM: An authoring aid for the PLATO IV CAI system (HumPRO RP-ED-75-6)", *Human Resources Research Organization*, Alexandria, Va, (April 1975).
- [10] Shu, N. C., et al., "Specification of Forms Processing and Business Procedures for Office Automation," *IEEE Transactions on Software Engineering*, Vol. SE-8, No. 5, (Sep. 1982) pp. 499-512.
- [11] Shu, N. C., "A Forms-oriented and Visual-directed application development system for non-programmers," *First IEEE Computer Society Workshop on Visual Language*, (Dec. 1984) pp.162-170.
- [12] Shu, N. C., "FORMAL: A Form-Oriented, Visual-Directed Application Development System," *IEEE Computer*, Vol. 18, No. 8, (1985) pp. 38-49.
- [13] Smith, D.C., Irby, C. and Kimball, R., "The Star user interface: An overview", *Proceedings of National Computer Conference*, (1982) pp. 515-528.
- [14] Takahiro Sato, "The S-P Chart Analysis", *Invited paper presented at the Seminar on CAI*, Taipei, Taiwan, (April 1986).
- [15] Yoshimoto, I., Monden, N., Hirakawa, M., Tanaka, M., and Ichikawa, T., "Interactive Iconic Programming Facility in HI-VISUAL", *Second IEEE Computer Society Workshop on Visual Language*, (June 1986) pp.34-41.
- [16] Zloof, M. M., "Query-by-Example: A Data Base Language," *IBM System Journal*, Vol. 16, No. 4, (1977) pp. 324-343.
- [17] Zloof, M. M., "QBE/OBE: A Language for Office and Business Automation," *IEEE Computer*, Vol. 14, No. 5, (1981) pp. 13-22.