

# Information Acquisition Model of Highly Interactive Tasks

**Takehiko Ohno**

Communication Science Laboratories,  
NTT, Japan  
*takehiko@rudolph.brl.ntt.co.jp*

**Hidemi Ogasawara**

School of Computer and Cognitive Science,  
Chukyo University, Japan  
*hidemi@sccs.chukyo-u.ac.jp*

## ABSTRACT

In this paper, we present a cognitive model of the information acquisition process in highly interactive tasks. The model is based on GOMS and extends the perceptual operation necessary to encode the visual information from the external world. To build the model, we observed users' behavior in an interactive task. "Pac-Man", a real-time computer game, was chosen as an example of the task. The results of the analysis confirm that the proposed model can explain users' behavior in real-time, time-constrained environments.

## 1 Introduction

The goal of this work is to provide a cognitive model that has the potential to estimate the performance of a human in performing a computer operation task, particular emphasis is placed on *bidirectional interactive tasks* (BDI tasks) between users and computers. By bidirectional, we mean interactive tasks in which the computer actively interacts with users by dynamically changing screen information, interface agents [4], sound, and the mix of media used. In BDI tasks, predicting the users' performance is very important. The complex interfaces demanded by the tasks are much more expensive to construct than simple traditional interfaces and it is also hard to modify their design afterwards. However, conventional techniques cannot adequately evaluate a design measure by predicting its usability, effectiveness, and acquisition costs. Models that help interface designers to create sophisticated interface designs are required.

GOMS [1] is a strong candidate to support the designing process. It can predict the performances of interactive tasks. It consists of four elements: *Goals*, *Operators*, *Methods* and *Selection rules*. *Goals* define the task status to be achieved. *Operators* describe the units of users' perceptual, motor and cognitive activities. *Methods*, procedures to accomplish a goal, are described using sequence of goals and operators. *Selection rules* are used when there are more than two candidate operators or methods. The power of GOMS is that the operation sequence is predicted using task knowledge, which is described by the rules of the task (task rule) and the task specification.

GOMS was originally used to represent the routine tasks performed by highly skilled users, e.g., editing tasks with

line editors and screen editors. In these tasks, the computer makes a response only if the user triggers some operations. We call these tasks *one-way interactive tasks* (OWI tasks). The task domain of GOMS has been expanded and now it covers bidirectional interactive tasks such as the long-distance telephone operators' tasks [2] and real-time video games [3].

The users' cognitive process in operating the interface differs between OWI and BDI. In OWI tasks, users set a goal and make a plan in the form of operation sequences to accomplish the goal. The goal does not change until the task is finished or unless there is some valid reason, e.g., the method chosen to accomplish the goal does not achieve the goal so that goal must be changed to select another method. Operations are selected according to the current goal and the status of the task.

In BDI tasks, on other hand, the information on the screen and the status of the task dynamically changes within a period of time. BDI requires users to change their operations, and even sometimes, their goals to follow the most recent status. Generally speaking, the current task status is determined by the symbols on the screen so that it is necessary for users to encode these symbols continuously and instantly. If the user cannot encode adequately comprehend the symbols on the screen, it is difficult to select an appropriate method to accomplish the task. Especially if the status changes in a short while, the number of symbols that can be encoded is limited. The relationship between visual information acquisition and operation selection is not yet well formulated in GOMS model and it is necessary to formalize the relationship to expand GOMS.

We propose a model of information acquisition, called *the Information Acquisition Model* (IA-model) that focuses on the symbol encoding process within an external environment, e.g. computer display. The model is based on GOMS and the perceptual operation used to encode symbols from the external world is extended. To build IA-model, we first observed a user's behavior in a BDI task. An eye mark recorder was used to measure the user's eye movements, they were used to determine the kind of symbol encoded by the user.

In this paper, we will first describe the BDI task, Pac-Man. Pac-Man is a popular real-time, maze-type video game. This will be followed by the results of the exper-

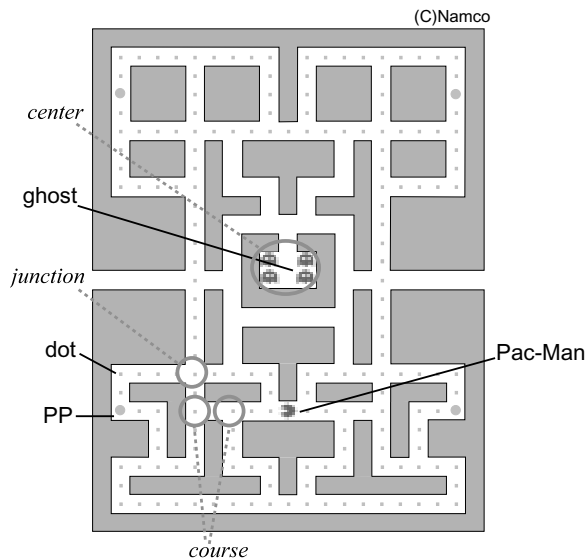


Figure 1: Screen image of “Pac-Man”. Pac-Man, ghost, PPs and dots are always displayed on the maze. *Courses*, *junctions* and *center*, which are used to analyze the frequency of symbol encoding, are determined by the Pac-Man’s current position and his direction.

iment. In the experiment, the subject performed the task more than 100 times and the end became an expert player. We will then introduce GOMS and also explain the detail of IA-model. Finally, we will discuss several factors that should be considered in modeling BDI tasks.

## 2 Pac-Man

“Pac-Man”, a typical maze-type real-time computer game, was used as a prototypical BDI task. Figure 1 shows a screen image of “Pac-Man” as seen in the experiment. It contains various bits of visual information, a Pac-Man, four ghosts, four Power Portions (PP) and a lot of dots arranged on the maze. Pac-Man, which is displayed in the center of the maze is the player’s avatar and is controlled by his keyboard operation. Four ghosts moving around the maze and they try to kill Pac-Man by catching him. When Pac-Man eats a PP, he acquires a power to kill ghosts for a short period. Players got bonus points by killing ghosts, the points are doubled for each additional ghosts killed in the same power-up time. When Pac-Man eats all the dots or is killed by ghosts for three times, the game is finished.

We developed a Pac-Man program that is able to record the sequence of players’ operations. Its rule is slightly different from the original version; in the original version, the game continues until all Pac-Man are killed but in our version it finishes once the first stage is cleared (all dots and PPs are eaten). No special fruits appear during the game (eating fruits added bonus points in the original version). The ghost’s movement algorithm, the speed of the ghosts and Pac-Man, and the power-up period also differed from those of the original version. Pac-Man is controlled through the keyboard (cursor key or ten key),

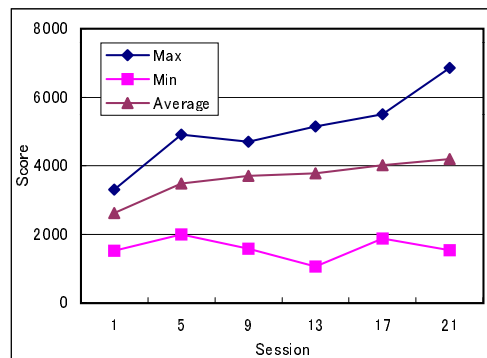


Figure 2: Subject’s score in each block of 4 sessions.

Table 1: Frequency of the symbol encoding operations observed in the experiment.

Session	Center	Course	Junction	Ghost	Total
1	0.02	0.92	0.69	0.40	2.03
5	0.09	1.20	0.99	0.63	2.90
10	0.07	0.97	0.53	0.87	2.43
15	0.07	0.92	0.57	1.00	2.56
20	0.12	0.60	0.52	1.18	2.42
24	0.11	0.82	0.49	0.55	1.96

not a joystick.

## 3 Experiment

We observed the user’s action, knowledge used, and visual information acquired in performing the BDI task. Visual information is inferred from the eye movement captured by the eye mark recorder.

### Procedures

The experiment consisted of 24 sessions. One or two sessions were done in one day and it took 1.5 months to finish the experiment. Each session contained three trial runs for practice and five game trials. In the trial runs, the participant controlled Pac-Man to eat all dots on the maze; there were no ghosts and no PPs. In the game trials, the participant was asked to maximize the score. After each set of two sessions, an information retrieval test was undertaken. In the retrieval test, information on the screen was made to disappear during the middle of the game and the subject was asked to note the position of the ghosts, Pac-Man, and PP upon a sheet of paper printed with the maze pattern.

### Participant

A male undergraduate participant was paid for his participation in the experiment. He was familiar with computer game but had played “Pac-Man” for only a few times and only knew the basic rules of “Pac-Man”.

### Materials

The experiment was undertaken on a Sun Workstation Sparc 10 (SS-10). The subject sat in front of a 19 inch

Table 2: Functional operators and methods used in the game. Methods with no \* mark are derived from the game instructions. Methods with \* marks are observed in the task and are not contained in the game instructions.

Operator	Method	Necessary symbols	Conditions of the selection rule
O-1: avoid-danger	O-1-1: escape-from-ghosts	course, ghosts	Ghost is near here, find the different way
	O-1-2*: move-with-ghosts	course, ghosts	Ghosts are following Pac-Man
	O-1-3*: stay-here	ghosts	Ghosts are coming close to Pac-Man
	O-1-4*: find-escape-way	course, ghosts	Pac-Man exists on the bottom part of the maze
O-2: eat-dots	O-2-1: eat-dots	course, dots	Dots exist in front of Pac-Man
	O-2-2: move-to-dots	course, dots	Player knows the course to the nearest dots
O-3: eat-PP	O-3-1: eat-a-PP	PP	PP exists in front of Pac-Man
	O-3-2: move-to-a-PP	course, PP	Player knows the course to the nearest PP
O-4: kill-ghosts	O-4-1: move-to-a-ghost	course, ghost	Pac-Man is in power-up mode
	O-4-2: catch-a-ghost	ghost	Pac-Man is in power-up mode, ghost is near Pac-Man
	O-4-3*: move-to-center	course, ghost, center	Ghosts exists in the center square
O-5: gather-ghosts	O-5-1*: wait-at-safezone		Pac-Man is in the safe zone, Ghosts are not near here
	O-5-2*: move-with-ghosts	course, ghosts	Ghosts are following Pac-Man
	O-5-3*: leave-dots	course, dots	There are only a few dots on the maze
	O-5-4*: stay-here		Ghosts are coming close to Pac-Man
	O-5-5*: move-to-ghosts	course, ghosts	The course to the ghost is clear

CRT color monitor and the distance between him and the monitor was approximately 60 cm. During the experiment, the keyboard operation and the change of game status (e.g. ghosts' movements) were recorded by the SS-10, and the subject's eye movement was recorded by an eye mark recorder, NAC EMR-NC. EMR-NC was developed by our group to study various aspects of human cognitive activities [6].

The "Pac-Man" program used in the experiment used a 20Hz game clock so Pac-Man and Ghost data recorded every 50 msec. The size of the maze was 27 blocks in height and 21 blocks in width. The size of each block was 20 by 20 pixels, and it took 5 cycles (250 msec) for Pac-Man and ghosts to move to the next block.

#### 4 Results

Figure 2 shows the score (maximum, minimum, and average) of block of four sessions. The maximum score of the last four sessions was about 2.1 times higher than that of the first four sessions. On the other hand, the minimum score of the last four sessions was similar to that of the first four sessions. This indicates that the subject sometimes did badly even though his skill had improved. The frequency with which each symbol was encoded in the experiment is shown in Table 1. The number of the encoding operations are counted by the eye mark data which was recorded by the eye mark recorder. Seven kinds of symbol appeared during the experiment

There are seven kinds of (see Figure 1). *Pac-Man*, *ghost*, *PP*, *dot*, *course* (maze walls), *center* (the position of the center square where ghosts are set at the beginning of the game) and *score* (displayed on the upper-right corner of the screen). The results showed that the total number of the symbol encoding events does not depend on the player's skill. On the average, the player looked at the screen about 2.3 times per second. That is, it took about 430 msec to

encode a symbol. The most frequently watched position was *courses*.

#### 5 Task Analysis with GOMS

GOMS can represent users' task knowledge using various grain sizes. This paper conducts two level analysis: the *functional level*, which examines users' functional activities, and the *keystroke level*, which examines users' motor operation to control keyboard. Each of them has the same grain size with the function-level operators (FLO) and keystroke-level operator(KLO) analysis[3] which was used to predict highly skilled player's operation in the real-time game, Super Mario Brothers 3. There are three kinds of operators used in the analysis; perceptual, cognitive and motor operators. Perceptual operators encode symbols from the screen. Cognitive operators carry out cognitive activities. Motor operators are used to control keyboard.

The framework of GOMS is shown Figure 4 (a). In BDI tasks, visual information displayed on the screen is acquired by perceptual operators and then, cognitive operators are executed. If there are several candidate cognitive operators, selection rules are used to select one of them. The condition part of the selection rules is compared with the current status of the game and then the action part of the selected rule is executed. Finally, a motor operator is executed to control Pac-Man by the keyboard.

##### Operator Set

The set of cognitive operators and methods used by the subject in the experiment is shown in Table 2. Necessary symbols appeared on the screen to accomplish the method is also contained in the set. Operators and methods are defined from the task instructions and the results of our observations. There are five important cognitive operators and 16 methods. Seven methods are taken from the game instructions and nine methods ( shown by the \* mark in

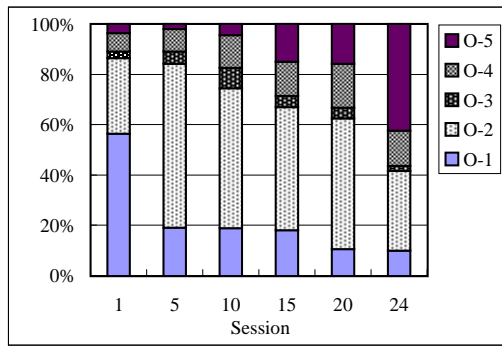


Figure 3: Distribution of five cognitive operators used in the second trial of each session 1, 5, 10, 15, 20 and 24.

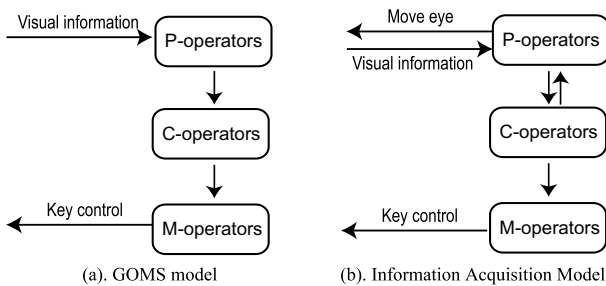


Figure 4: Framework of GOMS and the information acquisition model. P, C, M-operators means perceptual, cognitive and motor operators.

Cycle	P-operators	C-operators	M-operators
00 left	P:look-course	C-2-1:eat-dots	M:press-UP
25 up	P:look-course	C-2-1: eat-dots	M:press-LEFT
85 left	P:look-course	C-2-1: eat-dots	M:press-UP
105 up	P:look-course	C-2-1: eat-dots	M:press-right

Figure 5: Example of the task analysis by GOMS. Session 24, trial 2, 0-119 cycles.

Table 2) were observed in the real tasks.

Task analysis was conducted in the following manner:

- A task analysis unit is defined by player's each keystroke. Therefore, an unit contains one or more cognitive operators and one motor operator.
- Methods were determined by game condition which was acquired by perceptual operators.
- At the beginning of the analysis, methods derived from game instructions (no \* mark in Table 2) were used. The operations that were not explained by the existing methods were taken to be new methods.

We selected the second trials of sessions 1, 5, 10, 15, 20 and 24 for the task analysis. The ratio of cognitive operators used in the experiment is shown in Figure 3. At the beginning of the experiment, the participant used only known operators. In session 1, *O-1:avoid-danger* is

used more than 50% of the time. During the experiment, he gradually acquired additional operators and methods to get a high score. In session 24, the last session, *O-5:gather-ghosts* was used more than 40% of the time. It is necessary to use the methods in O-5 to achieve a good score.

Figure 5 shows an example of the operator sequence, the second trial in session 24. In the beginning of the trial, Pac-Man moved to the upper-left corner of the maze. *O-2-1:eat-dots* was used to eat dots. This analysis is not enough to explain the subject's behavior. For example, in Figure 5, the kind of the perceptual and cognitive operators are same, and it is not enough to determine the adequate motor operators. We need more strong framework when we analyze human's behavior in the highly interactive BDI tasks.

## 6 Information Acquisition Model

To explain human's behavior in the highly interactive BDI tasks, this section proposes IA-model which introduces eye movement operators into GOMS. The framework of IA-model is shown in Figure 4 (b). In the model, eye movements were executed by perceptual operators. When it is necessary to get the information on the screen, a perceptual operator that specifies the position of the screen explicitly, is used, and symbols displayed at the specified position were encoded.

The function of the perceptual operator in IA-model is similar to that in GOMS. Perceptual operators in GOMS are also used to encode symbols from the external world. The difference is that, in IA-model, the position that users want to look at is specified explicitly in the perceptual operators. Therefore, players are able to identify whether something is displayed or not on the specified position of the screen. On the other hand, they can not encode symbols if they does not look at the position explicitly. In "Pac-Man", a perceptual operator encodes the direction of the course, and the existence of ghosts, PPs and dots. Next, selection rules are used to choose the cognitive operator. Some cognitive operators execute motor operators for Pac-Man control.

### Task Analysis with the Information Acquisition Model

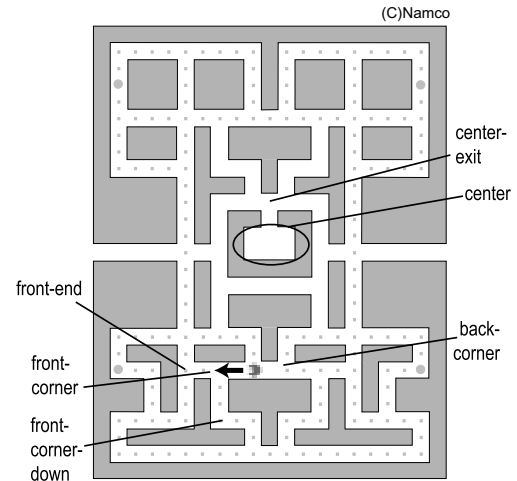
Examples of the perceptual operators observed in the experiment are shown in Table 6(a). The relative position from Pac-Man is used to specify the position of the player's gaze. Figure 6(b) shows which part of the screen the player looked at using perceptual operators. *Center*, *center-exit*, and *score*(which is not shown in Figure 6(b)) have fixed positions on the screen while the remainder have position relative to the current. The analysis process was the same as in GOMS.

### Skill Difference

In the experiment, as we mentioned in Section 4, executing a perceptual operator took a long time, more than 400 msec. In the highly interactive tasks, it becomes strong restriction for controlling. For example, the number of

Operator	Position of the eye mark
here	The exact position of Pac-Man
front-corner	The next corner in front of Pac-Man
front-corner-left,right, up,down	The next left/right/up/down corner from the <i>front-corner</i>
front-end	The end of the course Pac-Man is moving
back-corner	The next corner behind Pac-Man
center	The center square of the maze
center-exit	The exit way of the center square
score	Upper-right corner of the game screen where score is displayed

(a). Operator name. Center, center-exit and score are the absolute position of the screen, and the rest are the relative position from Pac-Man.



(b). Example of the perceptual operators when Pac-Man is moving from right to left.

Figure 6: Examples of the perceptual operators observed in the experiment.

perceptual operators that can be used to determine the motor operator depends on the course. When Pac-Man moves along a long course, there is enough time to encode many symbols displayed on the screen. On a short course, it is impossible for the players to encode many symbols. For example, in the shortest course, which consisted of two blocks, it took 10 game cycles (500 msec). In the case, players can encode only one or two symbols to determine the next direction. Players sometimes have to select a motor operator using few symbols.

In highly interactive BDI tasks, it is necessary for users to know which part of the screen they should look in the limited time. That is, the role of selection rules in choosing the cognitive operators that will invoke perceptual operators, is important. That is, the skill difference will be explained by the difference of the selection rule which is used to choose perceptual operators. Comparison of session 1 and session 24 clearly shows the difference in perceptual operators used by the subject over the course of the experiment.

Figure 7 shows a fragment of the analysis in the first part of the game (about 50 game cycles, 2.5 sec) in the second trial of session 1 and session 24. Candidates of motor operators and the encoded symbols of each motor operator are also shown. In session 1, the player mostly encoded symbols from in front of Pac-Man, whereas in session 24, he placed more emphasis on the center square. The symbols in the square are important at the beginning of the game because it is possible to infer the number of ghosts walking around the maze. If ghosts are in the square, it is safe to go to anywhere in the maze. The skill difference is not seen clearly in the frequency of the symbol encoding operations listed in Table 1. But it appears clearly, however, with very fine grain analysis which is shown in Figure 7.

### The Effect of Game Speed

Unlike the original “Pac-Man”, the game cycle in the experiment was fixed at 20Hz. In the original program, game speed increases gradually as the player clear more stages. IA-model will allow us predict the players’ behavior as game speed increases and the time available for information encoding is decreased.

## 7 Discussion

IA-model is useful to understand users’ behavior in detail. Here, we will discuss the possibility to use it to for the task evaluation and task prediction.

### Task Prediction

Task prediction by IA-model will be the next stage of this study. The strong power of GOMS is its ability in task prediction and IA-model has the same ability. In GOMS, a cognitive operator is proposed when certain elements of visual information are displayed on the screen. On the other hand, in IA-model, the encoding process by perceptual operators is necessary to propose a cognitive operator. Even though the task knowledge is defined as a set of cognitive operators and selection rules, the visual information needed to test the selection rule is not enough if perceptual operators do not encode symbols. This features will be useful in predicting mistakes in a play and task difference between users of different skill.

### The Difference Between GOMS and IA-model

There are two advantage of IA-model over GOMS:

1. Task analysis. IA-model can identify the symbols used in selecting operators in more detail. It is effective to understand users’ behavior, particularly novice users’ operations.
2. Task prediction. GOMS is a model for skilled users. On the contrary, IA-model can be applied to novice users who do not have the cognitive operators to select necessary perceptual operators.

