BIGexplore: Bayesian Information Gain Framework for Information Exploration

Kihoon Son*
Design Informatics Lab, Hanyang University
Seoul, Republic of Korea
gnswn000@gmail.com

Kyungmin Kim*
Design Informatics Lab, Hanyang University
Seoul, Republic of Korea
kkmkkmkkm86@gmail.com

Kyung Hoon Hyun†
Design Informatics Lab, Hanyang University
Seoul, Republic of Korea
hoonhello@gmail.com

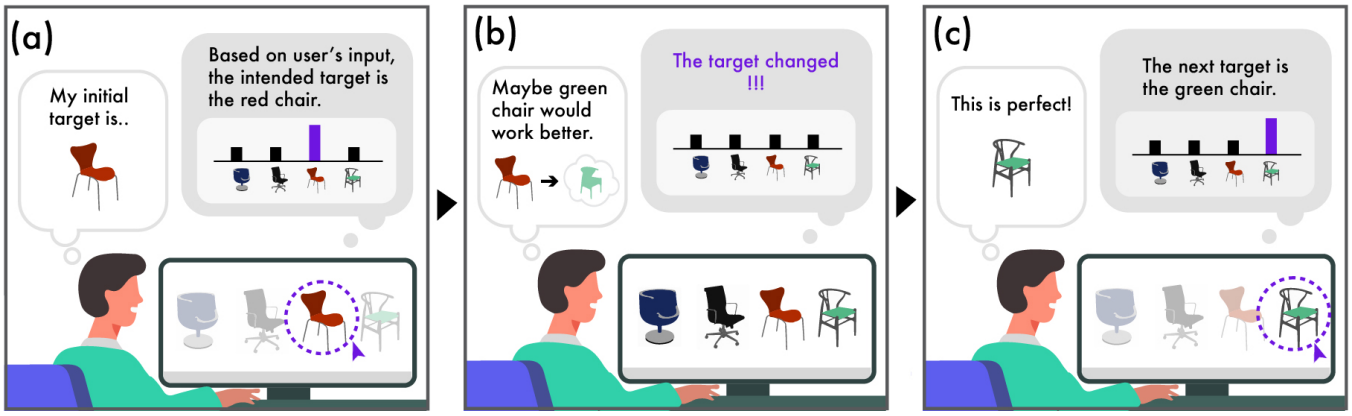

Figure 1: *BIGexplore* framework overview: (a) A user is searching for a chair design with an initial design objective. *BIGexplore* predicts the user's target chair; (b) The user updates the design objective after seeing the initial target chair. *BIGexplore* detects the user's target change point; (c) *BIGexplore* predicts the user's newly refined design objective.

ABSTRACT

The Bayesian information gain (BIG) framework has garnered significant interest as an interaction method for predicting a user's intended target based on a user's input. However, the BIG framework is constrained to goal-oriented cases, which renders it difficult to support changing goal-oriented cases such as design exploration. During the design exploration process, the design direction is often undefined and may vary over time. The designer's mental model specifying the design direction is sequentially updated through the information-retrieval process. Therefore, tracking the change point of a user's goal is crucial for supporting an information exploration. We introduce the *BIGexplore* framework for changing goal-oriented cases. *BIGexplore* detects transitions in a user's browsing behavior as well as the user's next target. Furthermore, a user study on *BIGexplore* confirms that the computational cost is significantly reduced compared with the existing BIG framework, and it plausibly detects the point where the user changes goals.

*Co-first author.
†Corresponding author.

This work is licensed under a Creative Commons Attribution International 4.0 License.

CHI '22, April 29-May 5, 2022, New Orleans, LA, USA
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9157-3/22/04.
https://doi.org/10.1145/3491102.3517729

CCS CONCEPTS

• **Human-centered computing** → **Interactive systems and tools**; • **Information systems** → **Users and interactive retrieval**.

KEYWORDS

Bayesian information gain, information exploration, design exploration, information retrieval, computational interaction

ACM Reference Format:
Kihoon Son, Kyungmin Kim, and Kyung Hoon Hyun. 2022. BIGexplore: Bayesian Information Gain Framework for Information Exploration. In *CHI Conference on Human Factors in Computing Systems (CHI '22), April 29-May 5, 2022, New Orleans, LA, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3491102.3517729

1 INTRODUCTION

Information exploration is a prerequisite for effective design decision making [7]. In the early phase of a design, designers focus on acquiring the design information to gain inspiration or develop a design [4]. In particular, designers look for images that provide information such as market trends or competing products to determine the overall design direction [12]. Information exploration during the design process is used to determine that the design direction consists of not only of goal-oriented actions, such as looking for a specific target direction, but also of sequentially connected actions for specifying the target direction(s) or looking for alternatives. In other words, a design exploration can be understood as a

complex cognitive process that results in a continuous update of a designer's design goals through sequential data acquisition and interpretation.

Researchers have extensively studied information-retrieval processes to support design exploration. Swire [11], C-Space [25], and VINS [4] provide the user with similar design information as they input the design features of interest. Matejka et al. [22] proposed a Dreamlens system that maps massive design images to the information space with the design features as the respective axes, allowing designers to explore the design space more effectively. Similarly, Hyun and Lee [12] also provided the user with a design space that, with each design feature as the axes, generates a specific design if the user clicks a point of interest in the design space. In the previous literature, the systems provided the user with design information as they enter the user input, making them passive learners. However, in design exploration, the design goal(s) may change or remain undefined. Therefore, the role of the exploration support system is to continuously update the designer's mental model by providing a series of feedback based on sequential user actions to specify the design goal(s). However, an interaction design that updates the user's mental model sequentially has yet to be realized.

In a complex information retrieval process, the system needs to understand the user's needs in real-time and provide information accordingly [23]. As the support system updates the user's mental model, the user is provided with real-time support for browsing or specifying the design goals. In this sense, the interaction between the system and the user is crucial [3], and interaction techniques based on computational methods have recently attracted significant interest in the HCI community. The Bayesian Information Gain (BIG) framework [19] is a computational interaction design based on the Bayesian experimental design, which combines Bayes' theorem and information theory. In the BIG framework, the system iteratively interacts with the user by updating its belief in the information space based on the current user action and by providing a view accordingly. Because these iterative processes stochastically quantify where the user's intended target is and support the user in achieving their goal, the BIG framework has been applied to target navigation situations [19, 21]. The BIG framework sequentially adapts itself to the individual browsing behavior of the user, making it suitable for the design exploration process where the nature of the designer's search varies.

However, the following limitations of the BIG framework must be solved to support the information exploration scenarios where the user does not know the location of the target and the intended target is changed. First, the BIG framework cannot provide real-time feedback in a large information space, owing to an excessive computational burden. Although Liu et al. [21] introduced the BIG-FileFast algorithm to solve this problem, it requires hierarchical data structures such as file data. Second, in certain situations, the BIG framework provides discontinuous views to the user. The BIG framework provides the user with the next view, which is expected to provide the system with the maximum information on the information space as the user takes the next action. In the process of providing the next view, the system calculates the information gain (IG) of all possible views at once. Therefore, the next view, which is an optimal view from the perspective of the system, may seem arbitrary to the user, as evidenced in an in-depth interview conducted after the experiment by Liu et al. [19]. Third, the BIG framework only focuses on cases where the intended target of the user is single, fixed, and predetermined. There is no method in the BIG framework for detecting the target change point when the user explores another target after selecting an initial target. As mentioned in [20], incorporating pure exploration situations of changing targets or more than one target will expand the BIG framework into a more general framework that efficiently supports the information exploration.

Therefore, we introduce the *BIGexplore* framework, a generalization of the BIG framework to support the user's information exploration process. The *BIGexplore* can provide a continuous next view based on the user's search action by employing the *Sequential_Search* algorithm, and it incurs a low computational cost. Also, as illustrated in Figure 1, the *BIGexplore* framework can predict the user's newly refined design objective by detecting the user's target change point by applying the *Initialize_Detect* algorithm. The remainder of the paper is structured as follows. After a review of related work, the *BIGexplore*'s exploration scenario, interface, and two main algorithms are explained. We then introduce four different frameworks used in the experiments: *non-BIG*, *BIGbase*, *semi-BIGexplore*, *BIGexplore*. In the user study section, three different comparative experiments are conducted to validate the *Sequential_Search* and *Initialize_Detect* algorithms, which constitute the *BIGexplore* framework. Finally, the possibilities of applying *BIGexplore* to various domains as interaction designs are discussed.

2 RELATED WORKS

2.1 Design Exploration and Information Retrieval

Several retrieval systems have been introduced in various domains to support designers in the early phase of design reference image exploration. Swire [11] and VINS [4] have been introduced as graphical user interface (GUI) retrieval systems, suggesting similar designs to the GUI made by the user. Similarly, to support a design exploration, Sketchplore [26] generates various layout designs based on user sketching by inferring the design task. In industrial design, Matejka et al. [22] proposed a Dreamlens system that suggests desk designs similar to that of the user's interest. Dreamlens allows users to explore desk designs by customizing desk features. Son et al. [25] proposed a C-Space system that provides the most analogous floor plan design when the user provides the system with the floor plan information. Parallel design exploration systems have also been used in spatial designs [9]. Recent design exploration support systems are designed to provide design information to users based on the user input. However, recent systems have made users passive in the sense that they are unilaterally provided with design information from the system. To support designers in exploring a massive design space for specifying the design goal(s) by providing real-time feedback, an interaction design that enables an appropriate human-system interaction is necessary.

Similarly, Shen and Zhai [23] insisted that the information retrieval process should understand the user's needs and then provide information through interactions with the user. Taking this a step further, we believe that systems that support information exploration should help users reach their goals by providing interactive

feedback based on user input. Accordingly, numerous studies have been conducted by assigning different levels of importance to the information depending on the user's exploration patterns. Fitchett et al. [8]'s file navigation interface uses the AccessRank prediction algorithm to highlight folders that can be the user's target and suggests the expected navigation path for the target. Hendahewa and Shah [10] introduced a system suggesting an appropriate search path for the user based on the number of clicks and the time spent on a given web page visited by the user. Dynamic maps [15] were proposed to support users in exploring the image space by providing similar images based on the user's panning actions. Besides, Sherkat et al. [24] made appropriate clusters of documents available to the user by providing confidence levels for the key terms that the user considered are important. However, information retrieval systems that (1) predict the user's target based on the user action and (2) detect the transition of the target are yet to be developed. *BIGexplore* is an information retrieval framework that accommodates these two features to support more effective information exploration in the changing goals. Specifically, by adopting the BIG framework, the *BIGexplore* framework provides the user with user-action-based feedback to sequentially update the user's mental model, enabling the user to specify the target.

2.2 Bayesian for Optimization

Bayesian optimization is a design strategy for optimizing the design and model parameters [17]. Bayesian optimization can be conceptualized as a two-fold strategy. Once an objective function is suggested by the system, numerical optimization is performed to provide an optimal point for the model parameters in the information space. In the HCI field, it has been used to sequentially improve the design and model parameters with human(s)-in-the-loop [6, 13, 14, 16, 18]. It is expected to become more popular in HCI because the objective function defined in the information space is unknown. However, in situations such as design exploration where the parameter(s) to be optimized is(are) unclear [1], applying Bayesian optimization may be difficult. The BIG framework [19] is slightly different. The objective function is no longer a black box: the expected IG can be understood as an objective function.

The BIG framework, based on the Bayesian experimental design (BED), has been introduced in HCI as a computational interaction design between the computer and the user with the purpose of finding the prespecified target. The BIG framework can be understood as an iterative three-step process, and the process continues until the user achieves their goal. The three-step process described by Liu et al. [19] is as follows:

(1) The system quantitatively interprets the intention of the user action from a probabilistic standpoint based on the user behavior model, likelihood.
(2) The system updates its belief regarding the information space using Bayes' theorem.
(3) The system presents the user with the next view under the maximization of the expected IG criterion.

However, as Liu mentioned in Bi et al. [3], the BIG framework incurs a higher cognitive load for users, which leads researchers to consider more balanced interaction and shared control by leveraging the expected IG. A higher cognitive load occurs because the expected IG criterion is solely for the maximal reduction of the system's uncertainty in the information space. Mathematical optimality does not always imply the optimality for the user. One of the key points to consider when applying computational techniques to an interaction design is to ensure a smooth coevolution of the user and the system to a state of improved interaction [3]. Thus, in *BIGexplore*, the system ensures a smoother interaction between the human and the machine by providing the user with a series of user-input-dependent sequential views. *BIGexplore*'s *Sequential_Search* algorithm adopts the expected IG criterion from a restricted number of possible views that seem to be relevant to the current action phase of the user (a detailed explanation is provided in Section 3.3.1.). In other words, the system considers both the mathematical optimality and the current behavior of the user. Additionally, the BIG framework suffers from a computational burden as the size of the information space increases. To circumvent the excessive computation time when applying the BIG framework to the file retrieval process on a database of tree structures, the BIGFileFast algorithm [21] was proposed. Although it is an efficient file search algorithm for hierarchical structures, the BIGFileFast algorithm does not suit sequential data structures as in the design exploration process. Finally, the BIG framework proposed by Liu et al. [19] and its applications [19, 21] support only goal-oriented cases [20], which can be defined as cases that satisfy the following three conditions:

- Condition 1: The user has a prespecified target in mind and knows its location in the information space.
- Condition 2: The user looks for a single target in a given information retrieval process.
- Condition 3: The target does not change in that information retrieval process.

For simplicity, we define cases that violate at least one of the aforementioned three conditions as changing goal-oriented cases in the rest of the paper. In the design exploration process, where the designer is in the process of exploring an unspecified design direction, the assumption of finding a single goal is unrealistic [27]. Even if the designer has the target design direction in mind, the direction may vary as the exploration phase continues, or the designer may be looking for multiple target directions. *BIGexplore* makes the BIG framework applicable to changing goal-oriented cases by applying our *Initialize_Detect* algorithm (a detailed explanation is provided in Section 3.3.2.) that detects changes in the exploration phase of the designer.

3 BIGEXPLORE FRAMEWORK

3.1 Overview

In this section, we provide an overview and core of the *BIGexplore* framework. The information search scenario and the vanilla interface of the *BIGexplore* are also introduced. As mentioned in Section 2.2, *BIGexplore* adopts our newly proposed *Sequential_Search* algorithm, a constrained version of the expected IG criterion for a continuous interaction. Moreover, by employing the *Initialize_Detect* algorithm, *BIGexplore* supports changing goal-oriented cases by initializing a belief in an information space with a uniform distribution when it is determined that the user's browsing behavior has

changed. The two main algorithms are described in detail in this section.

3.1.1 Information Searching Scenario. *BIGexplore* focuses on an information exploration situation that is most closely related to the design reference image exploration. Because the user does not know the location of the target information in the changing goal-oriented case, the user engages in browsing behaviors to find the target. In this process, scanning the information space at the macro level and refining the information at the micro level are essential and natural browsing behaviors. The intended target that the user wants to see also frequently changes during the exploration process. Although the intended target may not be clearly defined, the user considers the intended target. Therefore, we hypothesized the following information searching scenarios in *BIGexplore*:

- The user knows the intended target, although the target may not be clear.
- The user cannot know the location of the intended target.
- The user selects a target by repeating the following actions.
  - Scanning the information spaces at the macro-level. (i.e., Moving to another search area to find more information)
  - Refining the information spaces at the micro-level. (i.e., Observing closely to determine whether the information is the intended target or not)

3.1.2 Vanilla Interface. The *BIGexplore* vanilla interface consists of three main components (Figure 2): the minimap is presented on the left, information space in the middle, and user command indicator on the right.

Information space. The information space is where the information exploration process is performed. In general, the user cannot specify the location of the target in the information retrieval. The information space of *BIGexplore* is randomly distributed without a specific dimension. If there are dimensions that can map information to a specific location within a space, the user can determine the approximate target location. For example, suppose that chair images are mapped according to the main color in a three-dimensional space composed of three RGB axes. If the intended target is a blue chair, the user is more likely to conduct the exploration action only in the area where the blue value is high. Thus, the system can more easily predict the target of the user. In other words, if a dimension exists in the information space, the user's searching action is mainly performed on the target location; thus the BIG framework performs a biased update. The BIG framework can be evaluated better than the actual performance owing to biased updates. This means that it is difficult to accurately analyze how well the BIG framework predicts the user's target only with the searching actions in the information space. Therefore, we designed an information space without dimensions in a vanilla interface (Figure 2-b).

Four user actions. Users can conduct four search actions within the *BIGexplore* framework: zoom-in (+), zoom-out (-), pan (↔), and click (∗). These four actions are also provided by default in a general image search interface. For example, Windows File Explorer provides a zoom action using the mouse scroll and control key within a folder. Mackintosh's Finder provides a panning and zoom action using a trackpad. Similarly, a zoom and scroll are provided for navigation in the image gallery of the smartphones. When searching for images on the web, the user selects the desired image result using the scroll and zoom functions. In the image search process, a panning action for scanning the image information, a zoom action for refinement, and a click action for selecting a target are general and fundamental actions. Therefore, *BIGexplore* utilizes these four actions in the image exploration process and highlights the corresponding action on the user command indicator (Figure 2-d).

Minimap and next view. After a certain loading time, *BIGexplore* provides the user with the next view of the information space and updates the minimap based on the current user action. The minimap visualizes the system's updated belief in the information space by providing different transparency levels (Figure 3-a). Images with higher probabilities are expressed more intensely. Apart from the transparency level, images within the next view are colored purple, and the others are colored light gray. Also, when *BIGexplore* provides the next view to the user in the information space (Figure 2-b), images included in the next view are drawn with a purple border, and those not included in the next view are faded out (Figure 3-b). The interaction process of the system providing the next view to the user is illustrated on Figure 3-b. An example of the minimap and the next view in the interface are illustrated on Figure 3-c. With the aid of the next view and the updated minimap, the user continues to explore the information space until reaches the target.

3.2 Core Components

3.2.1 Notations for the BIGexplore Framework. We now describe in detail how *BIGexplore* modifies the three-stage navigation process from the BIG framework [19] to accommodate information retrieval. First, we define the following.

- $\Theta = \{1, 2, \ldots, n\}$ where $n = card(\Theta)$ represents a discrete information space that consists of a set of all possible images that the user can choose. θ means an image, and the probability that it is an intended target is $P(\Theta = \theta)$.
- t refers to the number of iterations performed on a given information retrieval process.
- $x^{(t)}$, a subset of the information space Θ, is the view that *BIGexplore* provides to the user on the t-th iteration.
- $X^{(t)}$ refers to a set of all possible views x that the system can provide to the user as $x^{(t)}$.
- $y^{(t)}$ refers to a user command made in the t-th iteration. $Y^{(t)}$ refers to a set of all possible user commands that the user can make in the t-th iteration.
- $Y^{(t)}$ depends on $card(x^{(t)})$. Input space $Y^{(t)} = \text{Range}(Y^{(t)}) \subset \{+, -, \leftrightarrow, *\}$ changes sequentially, since some inputs are not possible given the current view $x^{(t)}$ as follows:
  - $\text{card}(x^{(t)}) = 1$: $Y^{(t)} = \{-, \leftrightarrow, *\}$;
  - $\text{card}(x^{(t)}) = n$: $Y^{(t)} = \{+, *\}$;
  - $1 < \text{card}(x^{(t)}) < n$: $Y^{(t)} = \{+, -, \leftrightarrow, *\}$.
- $y^{(t)}_{spec}$ refers to the remaining images on the screen after $y^{(t)}$.

3.2.2 Likelihood and Prior Modeling. Next, the browsing behavior of the user and the system's prior belief in the information space, Θ,

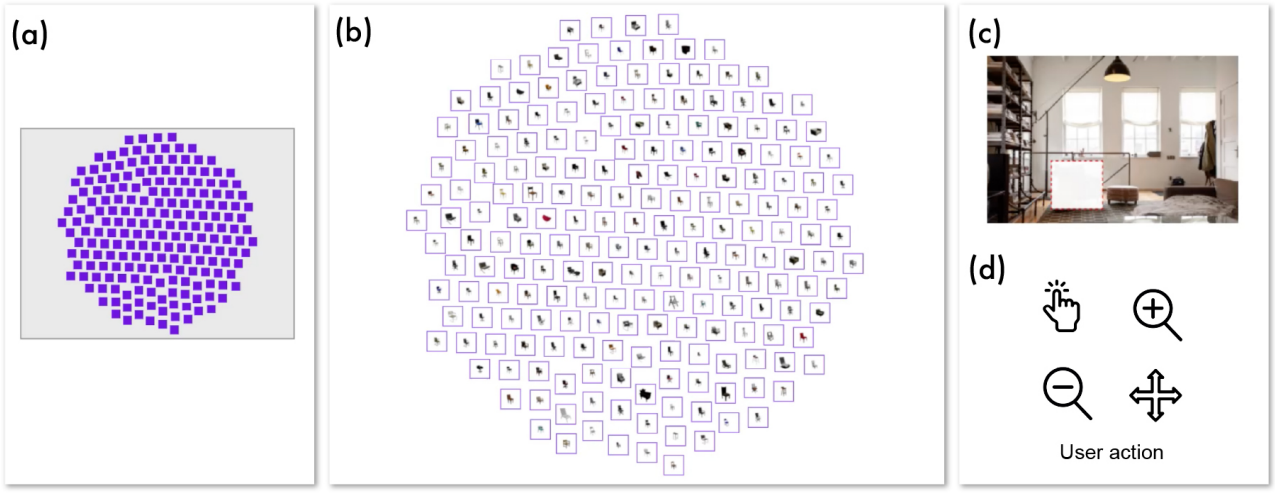

Figure 2: Vanilla interface of the *BIGexplore* framework: (a) minimap, (b) information space, (c) design objective, and (d) user action.

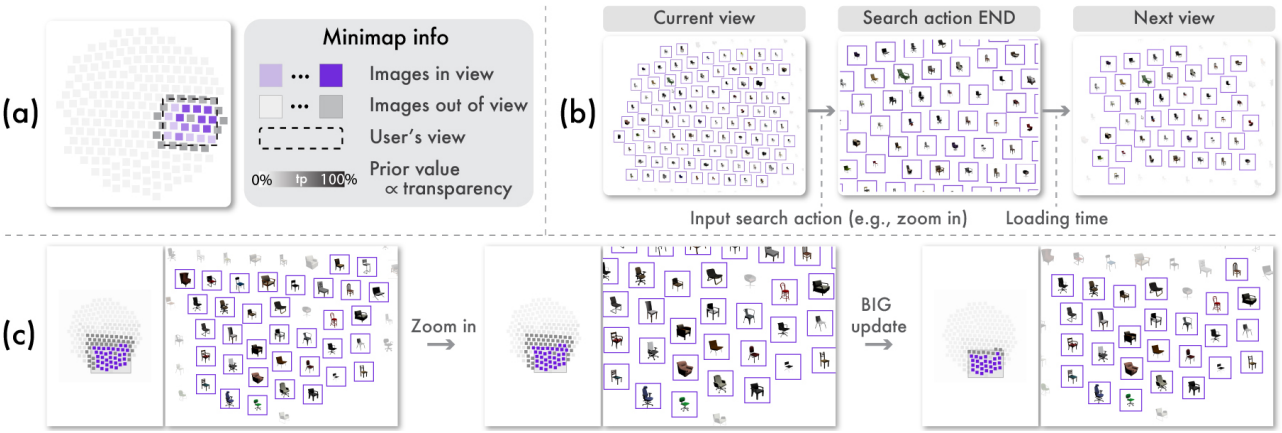

Figure 3: Example of the minimap and next view: (a) minimap, (b) next view providing process, and (c) actual examples in the vanilla interface.

must be specified before applying the BIG framework. Likelihood can be understood as the system's interpretation of the intention of the user command, $y^{(t)}$ [19]. The Bayes' theorem then combines likelihood and prior to update the system's belief in information space Θ, $P^{(t)}(\Theta = \theta)$. The system's initial belief on the information space Θ is set to a uniform distribution. The likelihood and prior are defined as follows:

Modeling the user behavior (likelihood). We have modeled the user behavior on the *BIGexplore* interface into six cases based on (1) whether the user's intended target $\theta^{(true)}$ is in $x^{(t)}$ and (2) $card(x^{(t)})$ (see Equations (1)-(6) in Appendix). We also assumed that a user command can be performed by mistake.

- User Model Defined by *BIGexplore*:
  - Zoom-in (+), zoom-out (-), pan (↔), and click (∗) are four possible user commands on the *BIGexplore* interface.
  - Click (∗) is possible regardless of the size of $card(x^{(t)})$ and of positional relationship between $\theta^{(true)}$ and $x^{(t)}$ if the user wants to click the image in the current view.
  - When $\theta^{(true)}$ is in $x^{(t)}$, zoom-in (+) and click (∗) commands are assigned with relatively high probabilities; zoom-out (-) and pan (↔) have relatively high probabilities for the opposite case.
  - The size of $card(x^{(t)})$ is considered while modeling user behavior. When $\theta^{(true)}$ is in $x^{(t)}$, for instance, the probability of zoom-in (+) is directly proportional and that of click (∗) is inversely proportional to the size of $card(x^{(t)})$, respectively. However, this is the case in which the $card(x^{(t)})$ is relatively large (Case 2). When $card(x^{(t)})$ is small, there

is an equal possibility that the user may zoom-in or click (Case 3).

Our proposed user behavior modeling on the *BIGexplore* interface is as follows:

- Case 1: When $\theta^{(true)} \in x^{(t)}$ and $card(x^{(t)}) = 1$, a probability of 0.9 is assigned to click ($*$). A probability of 0.05 is assigned to both (-) and pan ($\leftrightarrow$) actions, respectively (Equation (1) in Appendix).
- Cases 2, 3: When $\theta^{(true)} \in x^{(t)}$ and $1 < card(x^{(t)}) < n$, two different modelings are considered based on the size of $card(x^{(t)})$. The probability $\frac{card(x^{(t)})-1}{n}$ of zoom-in (+) is proportional to the size of $card(x^{(t)})$ and the other actions are assigned an equal probability of $\frac{n+1-card(x^{(t)})}{3n}$. However, this is the case in which the $card(x^{(t)})$ is relatively large (Equation (2) in Appendix). We considered $card(x^{(t)})$ is large if $\frac{card(x^{(t)})-1}{n} \geq \frac{n+1-card(x^{(t)})}{3n}$. When $card(x^{(t)})$ is small, the user zooming in as likely as they clicking (Equation (3) in Appendix).
- Case 4: When $\theta^{(true)} \in x^{(t)}$ and $card(x^{(t)}) = n$, a probability of 0.95 is assigned to zoom-in (+). Because $x^{(t)}$ is information space Θ itself, a probability of 0.05 is assigned to click ($*$) (Equation (4) in Appendix).
- Case 5: When $\theta^{(true)} \notin x^{(t)}$ and $card(x^{(t)}) = 1$, a probability of 0.9 is assigned to zoom-out (-). The pan ($\leftrightarrow$) and click ($*$) commands are understood as mistakes, and they are assigned a probability of 0.05 (Equation (5) in Appendix).
- Case 6: When $\theta^{(true)} \notin x^{(t)}$ and $1 < card(x^{(t)}) < n$, probabilities of 0.45 are assigned to zoom-out (-) and pan ($\leftrightarrow$), respectively. Zoom-in (+) and click ($*$) commands are understood as mistakes and are assigned a probability of 0.05 (Equation (6) in Appendix).

Interpreting user input. *BIGexplore* framework interprets user behavior in six changing goal-oriented cases. The probability distribution of the user behavior modeling represents the system's interpretation of user input. For example, when $card(x^{(t)}) = n$, the system believes that the user is most likely to zoom in (Case 4). If the intended target exists in the current view but $card(x^{(t)})$ is large, the user will have the highest probability of zooming in to see the target larger (Case 2). On the other hand, if $card(x^{(t)})$ becomes smaller, a high probability will be assigned to the click action (Case 3). Even when $card(x^{(t)})$ is 1 and the target exists in the view, our model considers a mistake and does not give a probability of 1 to the click action (Case 1). The system believes that the user would want to zoom-out to see other images if the user's intended target is not in the current view, and there is only one image (Case 5). If there are multiple images in the view without the target image, the system predicts a high probability of moving to another search area by zooming out or panning action (Case 6). In summary, *BIGexplore* considers $card(x^{(t)})$ and the user's mistake command for a proper interpretation of the user input in a large information space.

3.2.3 Expected Information Gain. By using the updated belief of the system on information space Θ at iteration t and the likelihood $P(Y^{(t+1)}|X^{(t+1)} = x, \Theta = \theta)$, the system calculates expected information gain $IG(\Theta|X^{(t+1)} = x, Y^{(t+1)})$ for all $x \in X^{(t+1)}$ and provides the user the view $x^{(t+1)}$ at iteration $(t+1)$, which is the maximizer of $IG(\Theta|X^{(t+1)} = x, Y^{(t+1)})$. The definition of $IG(\Theta|X^{(t+1)} = x, Y^{(t+1)})$ is as follows:

$$\begin{aligned} IG(\Theta|X^{(t+1)} = x, Y^{(t+1)}) &= IG(\Theta; Y^{(t+1)}|X^{(t+1)} = x) \\ &= \sum_{\theta \in \Theta} \sum_{y \in Y^{(t+1)}} P^{(t)}(\Theta = \theta, Y^{(t+1)} = y|X^{(t+1)} = x) \times \\ &\log\left(\frac{P^{(t)}(\Theta = \theta, Y^{(t+1)} = y|X^{(t+1)} = x)}{P^{(t)}(\Theta = \theta|X^{(t+1)} = x)P^{(t)}(Y^{(t+1)} = y|X^{(t+1)} = x)}\right) \end{aligned} \quad (1)$$

By definition, $IG(\Theta|X^{(t+1)} = x, Y^{(t+1)})$ is the average degree of dependence between two random variables Θ and $Y^{(t+1)}$ when x is presented as a view to the user at iteration $(t+1)$. From the perspective of the information retrieval process, $IG(\Theta|X^{(t+1)} = x, Y^{(t+1)})$ can be understood as the average decrease in the system's uncertainty in the information space at iteration $(t+1)$. One thing to note in the calculation of $IG(\Theta|X^{(t+1)} = x, Y^{(t+1)})$ is that $P^{(t)}(\Theta = \theta|X^{(t+1)} = x) = P^{(t)}(\Theta = \theta)$, considering the workflow of the BIG framework that provides $x^{(t+1)}$ using $P^{(t)}(\Theta = \theta)$.

3.3 Core Algorithms and the Hyperparameters

3.3.1 Sequential_Search Algorithm. We propose the *Sequential_Search* algorithm (Algorithm 1) that mediates the excessive cognitive load on the user and enables a continuous interaction between the user and the computer to increase the efficiency of the information retrieval process. The previous BIG framework [19] provides the next view with the maximum IG among all possible views, $X^{(t)}$. Thus, the next view may not be continuous to the user's search action. Furthermore, the next view can be interpreted as arbitrary from the user's point of view. To solve this problem, the *Sequential_Search* algorithm aims to provide a view with the maximum IG while continuous to the search action. To this end, the *Sequential_Search* algorithm provides the next view in two steps: 1) Create a set of views ($X^{(t+1)}$) that is continuous to the user's search action by considering $y^{(t)}$ and $y^{(t)}_{spec}$; 2) In $X^{(t+1)}$, a view with the largest expected IG is provided as the next view, ($x^{(t+1)}$). In the *Sequential_Search* algorithm, $X^{(t+1)}$ is dependent on the user commands $y^{(t)}$ and $y^{(t)}_{spec}$. Specifically, the *Sequential_Search* algorithm constrains the support $X^{(t+1)}$ of the objective function $IG(\Theta|X^{(t+1)} = x, Y^{(t+1)})$ based on the action behavior of the user at iteration t as follows:

- When the user zooms in, considering the user's convergent search behavior to take a detailed look at the subset $y^{(t)}_{spec}$ of $x^{(t)}$, $X^{(t+1)}$ is defined as the set of all the subsets of $y^{(t)}_{spec}$ of size $card(y^{(t)}_{spec}) - 1$.
- When the user zooms out, considering the user's divergent search behavior of taking a look at another region of information space Θ including $x^{(t)}$, $X^{(t+1)}$ is defined as the set of all the subsets of the information space Θ of size $card(y^{(t)}_{spec}) + 1$ having $y^{(t)}_{spec}$ as a subset.

- Because pan (↔) and click (∗) can be understood as actions in which the user's active intervention takes place, $y^{(t)}_{spec}$ is given as $x^{(t+1)}$ when the user pans or clicks.
  - Unlike the zoom-in (+) and zoom-out (-) actions where a subset of $x^{(t)}$ or $x^{(t)}$ itself is a subset of $y^{(t)}_{spec}$, the user can pan to any region of the information space Θ.
  - Clicking particular image on $x^{(t)}$ excludes all the images of $x^{(t)}$ except for $y^{(t)}_{spec}$.

Algorithm 1: *Sequential_Search*

Input: $y^{(t)}, y^{(t)}_{\text{spec}} = \{c_1, \ldots, c_k\}, P(Y^{(t+1)}|X^{(t+1)} = x, \Theta = \theta), P^{(t)}(\Theta = \theta)$
Output: $x^{(t+1)}$

$n := \text{card}(\Theta)$
$k := \text{card}(y^{(t)}_{spec})$
if $y^{(t)} = +$ **then**
  if *k=1* **then**
    $X^{(t+1)} = \{\{c_k\}\}$
  else
    Set of all possible ${}_k\text{C}_{\text{k}-1}$ = k subsets of $y^{(t)}_{spec}$ of images of size $k-1$,
    which is $X^{(t+1)} = \{\{c_1, \ldots, c_{k-1}\}, \ldots, \{c_2, \ldots, c_k\}\}$
else if $y^{(t)} = -$ **then**
  if *k=n* **then**
    $X^{(t+1)} = \{\{c_1, \ldots, c_k\}\}$
  else
    Set of all possible ${}_{n-k}\text{C}_1$ = n − k sets of size $k+1$ having $y^{(t)}_{spec}$ as a subset, which is $X^{(t+1)} = \{\{c_1, \ldots, c_k, c^*_1\}, \ldots, \{c_1, \ldots, c_k, c^*_{n-k}\}\}$, where $\{c^*_1, \ldots, c^*_{n-k}\} = \Theta \cap \left(y^{(t)}_{spec}\right)^c$
else
  $X^{(t+1)} = \{y^{(t)}_{spec}\}$
$x^{(t+1)} := \underset{x \in X^{(t+1)}}{\text{argmax}}\ IG(\Theta|X^{(t+1)} = x, Y^{(t+1)})$

3.3.2 Initialize_Detect Algorithm. In changing goal-oriented cases, Compared compared with goal-oriented cases where user commands, except user inputs made by mistake, converge to the user's intended target $\theta^{(true)}$ for all t, user commands show not only convergent search behavior to a particular image i but also divergent search behavior that explores the information space Θ in changing goal-oriented cases. The main problem in adopting a goal-oriented framework to changing goal-oriented cases as an interaction design between the user and the computer is updating the system's belief on information space Θ: there exist cases where $P^{(t)}(\Theta = \theta)$ "gets stuck". "Getting stuck" refers to situations where Bayes' theorem does not update $P^{(t)}(\Theta = \theta)$ well when the behavior of the user changes from convergent to divergent search process. Then the minimap will highlights the regions in which the user was previously interested. Therefore, we introduce our novel *Initialize_Detect* algorithm (Algorithm 2) detecting the changing point of the user's behavior from the convergent to divergent phase. *BIGexplore* initializes the system's belief on the information space Θ to a uniform distribution when the *Initialize_Detect* algorithm detects the case for a specified number of times, β. The *Initialize_Detect* algorithm believes that user behavior has changed from the convergent to divergent search phase in terms of two cases: consequential and behavioral aspects. In Algorithm 2, *max_prob_which* refers to the set of image(s) on the information space Θ with the maximum $P^{(t-1)}(\Theta = \theta)$. The consequential and behavioral aspects are as follows:

(1) Consequential aspects: Cases where the set of images $y^{(t)}_{spec}$ that the user has seen at the t-th iteration does not contain any images in *max_prob_which*;
(2) Behavioral aspects: Cases where $x^{(t)}$ contain at least one of the images in *max_prob_which* but the user zooms-out or pans.

Algorithm 2: *Initialize_Detect*

Input: $y^{(t)}, y^{(t)}_{spec}, P^{(t-1)}(\Theta = \theta), x^{(t)}$
Output: count_detect (`True` or `False`)

$max\_prob\_which := \{\theta \in \Theta \mid \underset{\theta \in \Theta}{\text{argmax}}\ P^{(t-1)}(\Theta = \theta)\}$
if $y^{(t)} = +$ or $y^{(t)} = *$ **then**
  if $max\_prob\_which \cap y^{(t)}_{spec} = \emptyset$ and $\text{card}(max\_prob\_which) < \alpha \times \text{card}(\Theta)$ **then**
    count_detect = `True`
  else
    count_detect = `False`
else
  if $max\_prob\_which \cap x^{(t)} \neq \emptyset$ and $\text{card}(max\_prob\_which) < \alpha \times \text{card}(\Theta)$ **then**
    count_detect = `True`
  else if $max\_prob\_which \cap y^{(t)}_{spec} = \emptyset$ and $\text{card}(max\_prob\_which) < \alpha \times \text{card}(\Theta)$ **then**
    count_detect = `True`
  else
    count_detect = `False`

3.3.3 Hyperparameters in the BIGexplore (α, β). In the proposed *BIGexplore* framework, we set the hyperparameters α and β as 0.2 and 3, respectively. First, α is a hyperparameter used to determine whether the $card(max\_prob\_which)$ value is significant in the information space. In the *Initialize_Detect* algorithm (Algorithm 2), $card(max\_prob\_which) < \alpha \times card(\Theta)$ indicates when the number of images currently having the maximum $P^{(t-1)}(\Theta = \theta)$ is less than $\alpha \times card(\Theta)$. If the information space with $card(\Theta) = 200$ is in a uniform distribution, $card(max\_prob\_which) = card(\Theta)$ means that the user's target is not specified at all. If α becomes too large, the *Initialize_Detect* algorithm will determine that the user's search phase changes even when there are a lot of *max_prob_which*. Conversely, if α is too small, the conditional statement is passed only

when $card(max\_prob\_which)$ is extremely small. By setting α to 0.2, we define a situation in which the user's target is specified only when $card(max\_prob\_which)$ was less than 20% of the entire information space.

Next, for predefined β as 3, the *BIGexplore* framework initializes the information space when the user changes the searching behavior three times. The first and second behaviors are assumed to involve leaving the area the user is currently focusing on. The behaviors after the second behavior are assumed to explore the next target. If β is 1 or 2, the framework is more likely to perform an incorrect initialization by reacting sensitively to the user's mistaken action. If β is 4 or more, the initialization may not be performed yet despite moving to the next target in the information space with $card(\Theta) = 200$. To avoid such situations in information spaces with $card(\Theta) = 200$, we set β to 3.

4 USER EXPERIMENT

Three different experiments were conducted to validate the two main algorithms (*Sequential_Search*, and *Initialize_Detect*) which comprise the *BIGexplore* framework in the changing-goal oriented case. Table 1 shows the conditions of each experiment briefly. The task and frameworks of each experiment are described in detail in the subsequent sections. All the experiments were conducted web-based system with the vanilla interface (Client framework: vue.js, hardware: Windows OS with an Intel® CoreTM i9 11900KF(3.5GHz), 128GB of RAM; Monitor: 24 Inch, 144 Hz, FHD; Server framework: Python flask server; Hardware: Linux OS with an Intel® CoreTM i7-7700 CPU, 64 GB of RAM).

4.1 Experiment Design

4.1.1 Frameworks to compare. Four different frameworks were used in the experiments (Table 2): *non-BIG*, *BIGbase*, *semi-BIGexplore*, and *BIGexplore*. We implemented the original BIG framework of Liu et al. (named *BIGbase*) [19]. The *BIGbase* framework, without the application of any algorithms proposed in this study, only shared the user behavior modeling. *BIGbase* and *semi-BIGexplore* were compared in Experiment 1 to validate the performance of the *Sequential_Search* algorithm. *Semi-BIGexplore* used only the *Sequential_Search* algorithm, and *BIGexplore* used both algorithms. These two frameworks were compared in Experiments 2 and 3 to validate the performance of the *Initialize_Detect* algorithm. *Non-BIG* is a simple zoom-and-pan search framework that does not provide a BIG update. The *non-BIG* framework was also compared with the proposed BIG frameworks (*semi-BIGexplore*, and *BIGexplore*) throughout the experiments.

4.1.2 Experiment Setting. Twenty-one participants (Age_{mean}=26.7; Age_{min}=23; Age_{max}=37; 12 males and 9 females) participated in the experiments. All the participants had normal or corrected-to-normal vision. The experimental procedure was conducted in two parts. First, the three experiments (Table 1) were conducted using *BIGexplore*, *semi-BIGexplore*, and *BIGbase*. Second, after three months, we conducted the three experiments again using non-BIG with the same participants to analyze the effect of the *BIGexplore* and *semi-BIGexplore*. Each experiment was finished when the user found all of the correct answers without a time limit. Every participant was instructed to participate in all three experiments, and no dropouts occurred, except for five participants who quit the task of finding the target images in Experiment 1 using *BIGbase*. To minimize the ordering effects, the order of the experiments, frameworks, datasets, and design briefs were randomly assigned. The target images were also randomly determined from all possible images. Each time the participants completed a given task using one of the frameworks, they participated in a NASA-TLX and satisfaction survey. In addition, an in-depth interview was conducted before the end of each experiment. The experiments took an average of 1.5 h per participant.

4.1.3 Experiment Objective. As shown in Table 1, three changing goal-oriented cases were designed. The objectives of the three experiments were as follows.

Experiment 1. Three different frameworks were compared in Experiment 1: *semi-BIGexplore*, *BIGbase*, and *non-BIG* (Tables 1 and 2). This experiment aims to determine how the *Sequential_Search* algorithm affected the user experience and task performance in the prespecified single-target search scenario. Participants were asked to find the correct SIGCHI logo from among 13 different images in the information space (Figure 4-a). A relatively small information space of $card(\Theta) = 13$ was considered to enable *BIGbase* to provide real-time feedback. When $card(\Theta) > 13$ was without the *Sequential_Search* algorithm, the loading time exceeded 24 s, making *BIGbase* an extremely inefficient system (Figure 5). Therefore, we made modifications to the SIGCHI logos, except for that in the answer, to encourage users to actively explore the information space (Figure 4-a).

Experiment 2. In Experiment 2, we aimed to evaluate the impacts of the *Initialize_Detect* algorithm. Thus, a prespecified multiple-target search scenario was designed to observe whether the *Initialize_Detect* algorithm appropriately detects the user's intended target by capturing the changing point of the user's browsing behavior. For the task in the experiment, participants were asked to find five prespecified human face images from 200 different human face images (Figure 4-b). We used open-source image data from Kaggle [5], which provides various human face images of different sexes, ages, and ethnicities, to create datasets having a size of 200 ($card(\Theta) = 200$). Five prespecified human face images are presented on the right side of the interface: when the user clicked the answer, a red border was drawn around the image in the information space, and the word "Find" was written above the answer on the right.

Experiment 3. In Experiment 3, we assumed unspecified multiple-target search scenarios in a changing goal-oriented case. Similar to Experiment 2, this experiment's objective was to evaluate the impact of the *Initialize_Detect* algorithm on an unspecified target search. The user's browsing behavior in the information space was different from that in Experiment 2 because no specified answers were provided. In Experiment 3, a design brief was given, and participants were asked to find a maximum of five chair images from 200 different options (Figure 4-c). We used the rendered chair images from Aubry et al. [2] to create two different datasets having a size of 200 ($card(\Theta) = 200$). We prepared two design briefs, each having an image with an empty box (in supplemented materials). The task of the participants were to choose a chair design that

Table 1: Each condition of the three experiments.

	Algorithm for Validation	$card(\Theta)$	Number of Answers	Frameworks to compare
Exp.1	*Sequential_Search*	13	1 (prespecified)	non-BIG; BIGbase; semi-BIGexplore
Exp.2	*Initialize_Detect*	200	5 (prespecified)	non-BIG; semi-BIGexplore; BIGexplore
Exp.3	*Initialize_Detect*	200	maximun 5 (unspecified)	non-BIG; semi-BIGexplore; BIGexplore

Table 2: Details of each framework used in the experiments (✓for with algorithm).

	non-BIG	BIGbase[19]	semi-BIGexplore	BIGexplore
BIG update	-	✓	✓	✓
Sequential_Search	-	-	✓	✓
Initialize_Detect	-	-	-	✓

Figure 4: Stimuli of (a) Experiment 1, (b) Experiment 2, and (c) Experiment 3.

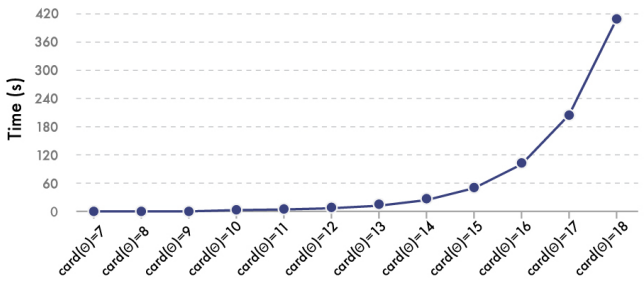

Figure 5: Computation times with respect to $card(\Theta)$.

stylistically fit the given image. The experiment ended when the participant found the item.

4.2 Results and Discussion

In this section, we provide an overview of the experiment results. After the overview, four main aspects of the results and real-world applications of *BIGexplore* are discussed.

4.2.1 Overview of the Results. Both algorithms (*Sequential_Search*, and *Initialize_Detect*) showed a significant performance in the changing goal-orientation cases. First, the *semi-BIGexplore* with the *Sequential_Search* algorithm provided better results than *BIGbase* in terms of the overall user experience and task completion time in Experiment 1 (Figures 5 and 6). In the case of *BIGbase*, five participants quit the task of finding the correct SIGCHI logo owing to the long system loading time and the next view providing time. Second, *BIGexplore*, which used the *Initialize_Detect* algorithm, outperformed *semi-BIGexplore* in terms of the target prediction. Although the search patterns of the user were different during both experiments, the *Initialize_Detect* algorithm predicted the user's intended target and the change point of the target. More details regarding the effectiveness of both algorithms are described in Sections 4.2.2 and 4.2.3. In Section 4.2.4, the differences between the non-BIG and BIG frameworks (*semi-BIGexplore* and *BIGexplore*) are described. Both *semi-BIGexplore* and *BIGexplore* significantly reduced the user's mental load and total number of commands compared with *non-BIG*. Furthermore, we analyzed IG values for each framework in Section 4.2.5. This section explains how BIGexplore effectively reduced the uncertainty of the information space Θ.

4.2.2 Impact of Sequential_Search Algorithm. In Experiment 1, the *Initialize_Detect* algorithm significantly affected the task completion time (Figure 7; $Mean_{BIGbase}$ = 248 s (SD = 65 s) and

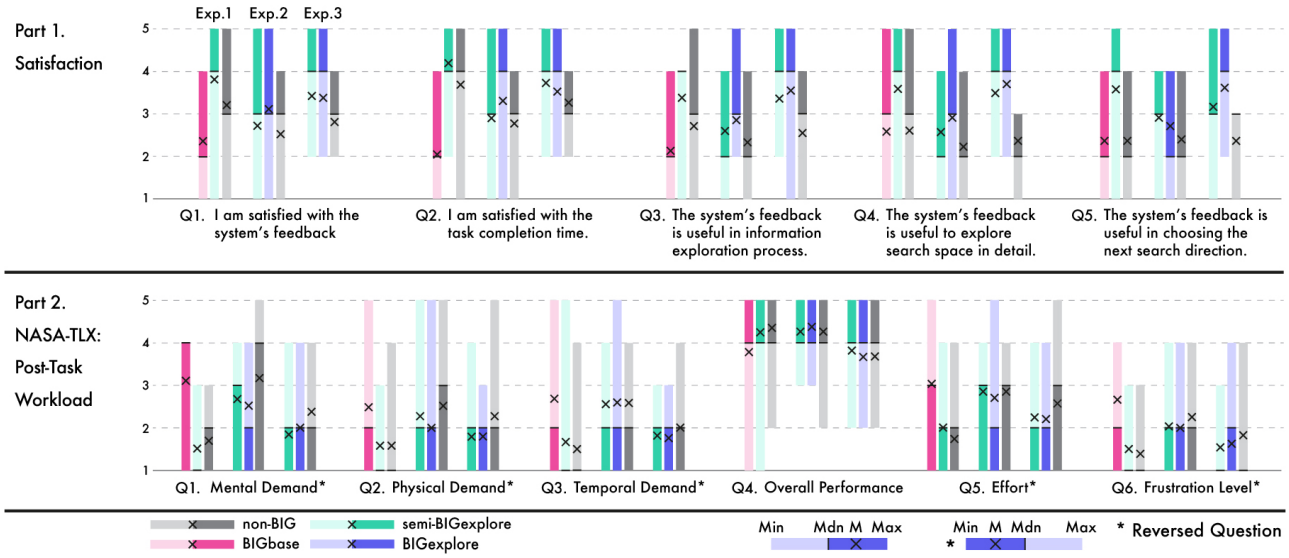

Figure 6: User experience results from the three experiments.

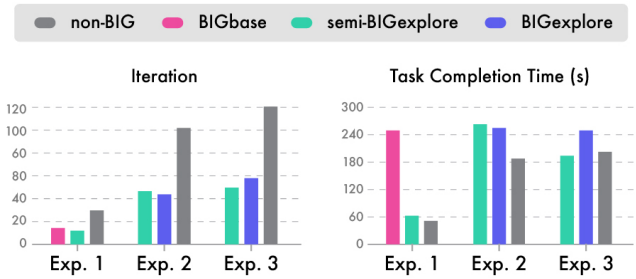

Figure 7: Iteration and task completion time results for the three experiments.

$Mean_{semi-BIGexplore}$ = 64 s (*SD* = 45 s); $p < 0.001$). Because *BIGbase* required excessive computations to provide the next view, the participants had difficulty receiving real-time feedback. A significant loading time before the next view was provided led to more stress and less satisfaction when using *BIGbase* as an interaction design. As observed in the responses from the NASA TLX worksheets (Figure 6), *BIGbase* requires high degrees of mental demand (3.14 (*SD* = 1.10) > 1.57 (*SD* = 0.68)) and physical burden (2.48 (*SD* = 1.31) > 1.62 (*SD*=0.75)) as well as duration (2.62 (*SD* = 0.99) > 1.71 (*SD* = 1.02) for single target searche (p < 0.05). In terms of the satisfaction with the task completion time, *semi-BIGexplore* was preferred over *BIGbase* (4.19 (*SD* = 0.86) > 2.05 (*SD* = 1.00); $p < 0.001$). Similar to the survey results, negative feedback on *BIGbase* was dominant in the in-depth interviews. Participants P3 and P4 responded that a long loading time frustrated them. In addition, P7 and P8 expressed a decrease in satisfaction owing to the long loading time, and pointed out that the delay in time also affected the exploration process. Most of the participants were satisfied with the fast next view feedback of the *Sequential_Search* algorithm.

The participants also revealed different levels of satisfaction with the next view. *Semi-BIGexplore* using the *Sequential_Search* algorithm provided a continuous view of the user's search action, whereas *BIGbase* provided the next view considering only IG. Five of the participants showed frustration in using *BIGbase* because they fell into in "got stuck" situation in a particular view, which was not the view of the images they were interested in. "Getting stuck" refers to a situation in which the BIG framework does not update $P^{(t)}(\Theta = \theta)$ when the intended target of the user changed. Thus, when a "getting stuck" occurred, *BIGbase* provided a fixed view regardless of the searching actions. The log result from P15 (Figure 8) demonstrates this case. In the in-depth interview, P14 responded that after clicking an incorrect answer, the system continued presenting the same image repetitively, regardless of their action. P21 had difficulty completing the tasks owing to the provisioning of views with unwanted images that were against the participant's commands. The results of the survey (Figure 6) also indicate dissatisfaction with the way *BIGbase* provided feedback compared with that for *Semi-BIGexplore* (Satisfaction Q1: $Mean_{BIGbase}$ = 2.33, $Mean_{semi-BIGexplore}$ = 3.81, $p < 0.001$; Q3: $Mean_{BIGbase}$ = 2.19, $Mean_{semi-BIGexplore}$ = 3.38, $p < 0.001$; Q4: $Mean_{BIGbase}$ = 2.57, $Mean_{semi-BIGexplore}$ = 3.62, $p < 0.05$; Q5: $Mean_{BIGbase}$ = 2.33, $Mean_{semi-BIGexplore}$ = 3.57, $p < 0.001$). Using *BIGbase*, participants had to put in more effort, but were more frustrated at the outcome (NASA-TLX Q5: $Mean_{BIGbase}$ = 3.05, $Mean_{semi-BIGexplore}$ = 2.0, $p < 0.001$; Q6: $Mean_{BIGbase}$ = 2.67, $Mean_{semi-BIGexplore}$ = 1.43, $p < 0.001$).

To quantitatively compare the appropriateness of the two different view provisioning methods, we analyzed all views provided by the system to determine whether the target image was within the view ($x^{(t)}$) during the last six iterations (Table 3). The percentage of the last six views containing the target image was 65.4% for *semi-BIGexplore* and 23.2% for *BIGbase*. The average sizes of the view during the last six iterations were 2.05 and 6.17 for *BIGbase* and

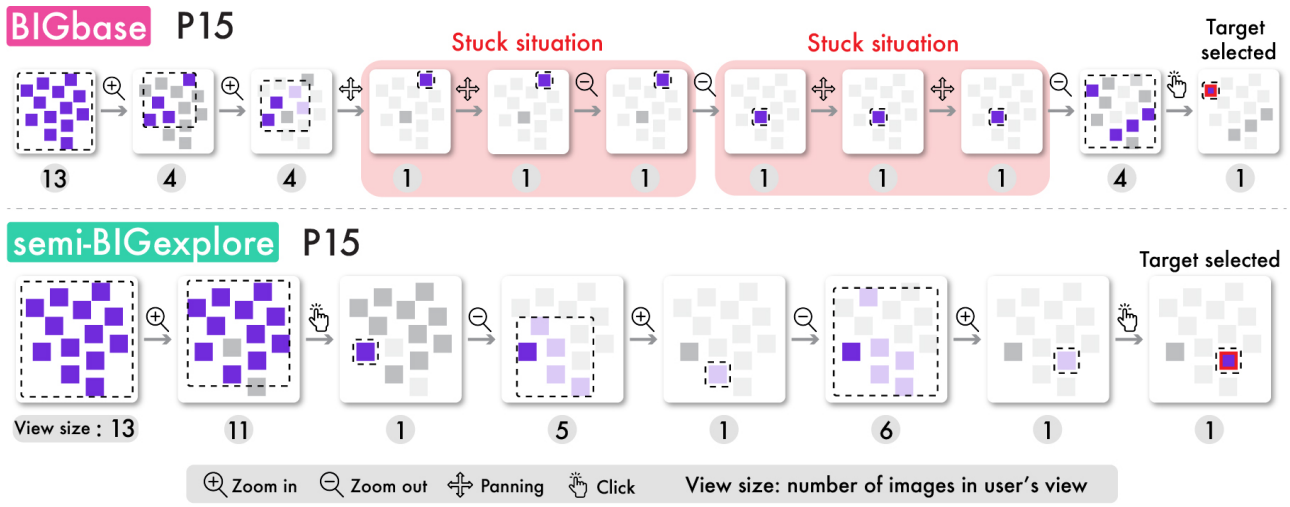

Figure 8: Log data of Experiment 1 using *BIGbase* (above) and *semi-BIGexplore* (below) frameworks.

Table 3: Summary results of the last six views. The second row represents the proportion of the last six views containing the target image for each framework; the third row shows the average view size of the last six views for each framework.

	BIGbase	*semi-BIGexplore*
Target in View	23.2%	65.4%
View Size (*Mean*)	2.05	6.17

Table 4: Results of Experiments 2 and 3 on the average of *max_prior_ratio* and *target_prediction_rate*

		semi-BIGexplore	*BIGexplore*
Exp.2	*max_prior_ratio*	2.6%	49.1%
	target_prediction_rate	11.4%	65.7% (6.6%)
Exp.3	*max_prior_ratio*	7.2%	47.3%
	target_prediction_rate	18.6%	68.7% (16.5%)

semi-BIGexplore, respectively. As shown in the log results of P15 (Figure 8), *semi-BIGexplore* using the *Sequential_Search* algorithm provides continuous views in terms of previous user actions and continuously includes the target image in the view. By contrast, *BIGbase* continuously provides the user with a view of size one with no target image (red boxes in Figure 8). This discontinuous and inaccurate provisioning of views would lead to a distrust in the system because the system would appear to be making a wild guess. In summary, the *Sequential_Search* algorithm contributed twofold to the exploration process: the task completion time and user experience. However, *Sequential_Search* ostensibly solved the problem of "getting stuck" by providing the next view related to the user's actions. To solve the problem of updating the prior for changing targets, the *Initialize_Detect* algorithm is required for the BIG framework.

4.2.3 Impact of Initialize_Detect Algorithm. Experiments 2 and 3 were conducted to validate the *Initialize_Detect* algorithm by comparing *BIGexplore* and *semi-BIGexplore* in different changing goal-oriented settings. Despite the different settings, the *Initialize_Detect* algorithm appropriately detected the user's current target by capturing the target changing point of the user's browsing behavior in both experiments. To quantify the performance of the *Initialize_Detect* algorithm, two evaluation indices were considered: (1) whether the probability assigned to the answer target was the maximum over the information space Θ when the participant had clicked that image (*target_prediction_rate*), and (2) the number of times the system assigned the changed target with the maximum probability until the participant clicked it (*max_prior_ratio*). The *target_prediction_rate* returned a value of "True" only if the maximum probability over the information space equaled the probability assigned to the current chosen target. By averaging the rate of the "True" count, we calculated the *target_prediction_rate* value in each framework. The *max_prior_ratio* represented to the number of iterations between clicking the (k)-th and $(k+1)$-th answers, assigning the largest probability to the $(k+1)$-th answer image for all values of $k = 1, 2, 3, 4$. Taking Figure 9 as an example, the *max_prior_ratio* values are $4/6$ and $0/7$ for the two log data.

In both experiments, the *Initialize_Detect* algorithm showed an outstanding performances in terms of the *target_prediction_rate* and *max_prior_ratio* (Table 4). The *BIGexplore* and *semi-BIGexplore* frameworks showed different updating behaviors for the system's belief in the information space when the target image is changed (Figure 9). Specifically, unlike *BIGexplore* that assigned the maximum probability to the next target since the system's belief on the information space has been initialized, the system's belief on the information space became stuck in the region near the previously chosen target in the case of *semi-BIGexplore*. This indicates that a stuck problem occurred when exploring a changing goal-oriented case using *semi-BIGexplore*. Table 4 presents this problem. The values

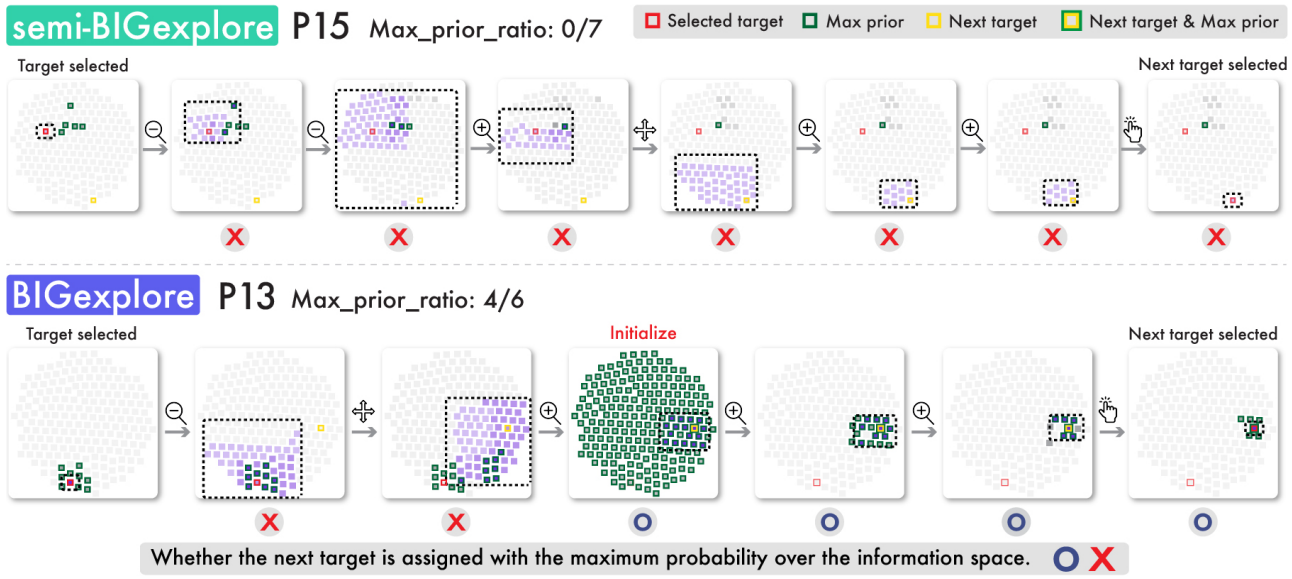

Figure 9: Log data of experiment 2 using *semi-BIGexplore* (above) and *BIGexplore* (below) frameworks.

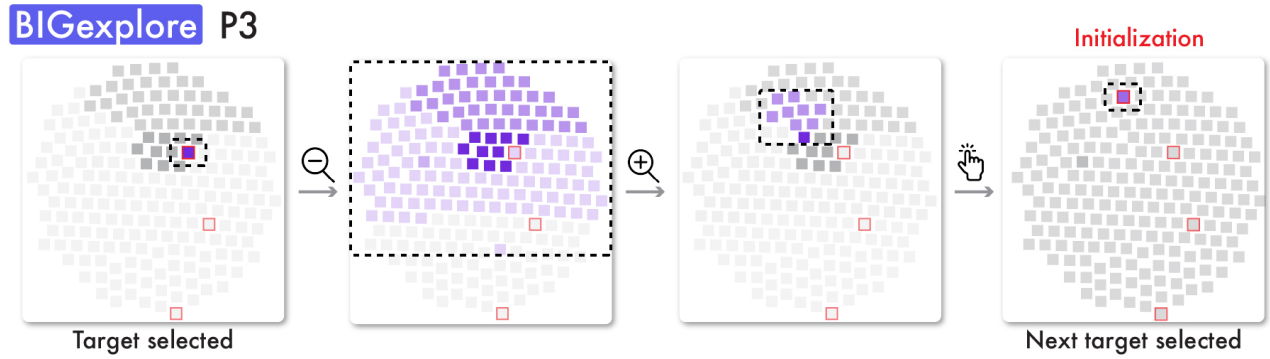

Figure 10: Log data of Experiment 3 where a late initialization has occurred.

of *semi-BIGexplore* in Table 4 indicate that the framework assigns a lower probability value to the user's intended target more accurately and more often in Experiments 2 and 3. A *max_prior_ratio* value close to 100% implies that the framework adapts appropriately to the change in the user's intended target by solving the "got stuck" situation. In this respect, the *Initialize_Detect* algorithm contributes to an accurate BIG probability update in the changing goal-oriented case. The values in parentheses in Table 4 refer to the case in which the initialization of the system's belief is conducted when the participant has clicked the changed target. These are cases in which the prior initialization is applied a step later and a difference exists between the values (Experiment 2, 6.6%; Experiment 3, 16.5% in Table 4). The moment of initialization is considered appropriate, at least when it has occurred before clicking the next target. The user behavior in Experiment 3 may be the reason for the slightly late initialization (Figure 10). Unlike the prespecified answer cases in Experiment 2 where the users click the target as soon as they find it in the view, in the unspecified answer cases, the participant may not click the intended target immediately even if it is provided in the current view. In this way, the participants memorize the location of the target chair in their minds and explore more designs. Then, when the participant cannot find a better chair for their task, they selected the memorized target (Figure 10).

4.2.4 Non-BIG vs. The Proposed BIG Frameworks. Unlike the proposed BIG frameworks, the *non-BIG* framework provides the result of the user's action input as it is. By comparing the *non-BIG* with the proposed BIG frameworks, we analyzed the contribution of the proposed BIG-based feedback (*BIGexplore* and *semi-BIGexplore*). The result of the comparison showed that the BIG frameworks had longer task completion time than the *non-BIG*. However, the number of user commands with *non-BIG* was significantly higher than those of the BIG frameworks. Specifically, both *BIGexplore* and *semi-BIGexplore* feedback reduced the number of user commands by up to three times in the all experiments (iterations in Figure 7; $p < 0.001$). Considering that the system loading time (response time between server and client) was included when measuring the task

completion time for the proposed BIG frameworks (Figure 7), it can be said that the BIG frameworks assisted the users to reduce both mental and temporal demands during the information exploration process. This was supported by the in-depth interview with the *non-BIG* users, *"I conducted many meaningless search actions, because I was conducting the task without the support of the system"* (P6–P7, P10, and P20). Specifically, in Experiments 2 and 3, where the information space was large, participants replied, *"Unlike the task with the BIG framework, experiment using non-BIG is too hard and frustrating"* (P1–P3, P6–P8, P10, P12, P15, P18, P20, and P21).

In the survey results (Figure 6), similar to the overall in-depth interview, the participants answered that *semi-BIGexplore* and *BIGexplore* were more useful than *non-BIG* in exploring (Satisfiaction Q3 and Q4) and setting the search direction (Satisfiaction Q5) in the information space. Specifically, the participants (P1–P3, P6, P7, P10, P12, P13, and P15) stated that, *"In Experiments 1 and 3, it was impossible to determine whether the image was correct or not immediately. Because it was a task to consider slight differences between images, the next view feedback of the BIG framework reducing the image options could be a significant support in setting the search direction."* However, in Experiment 2, no significant difference was observed between *non-BIG* and *BIGexplore* in determining the next search direction (Satisfaction Q5: $Mean_{non-BIG}$ = 2.33, $Mean_{BIGexplore}$ = 2.71, $p >$ 0.05). Participants (P4, P11, P14, and P17) described the reason as, *"Experiment 2 aims to find the answer among 200 images, so I set the search direction from when I started the task. Therefore, the feedback (of BIGexplore) did not significantly affect the search direction."* Unlike in Experiment2, participants stated that their search direction became meaningless because there were no prespecified targets (P5, P7, P10, P20, and P21) in Experiment 3. Specifically, the target search process using non-BIG in Experiment 3 was overwhelming to the participants (P5-P7, P10, P12, P14, P17, P18, P20, and P21). However, unlike in Experiment 2, the participants commented that *BIGexplore* (minimap and next view) assisted them to set the search direction. Thus, better satisfaction results were obtained for the *BIGexplore* than for the *non-BIG* in exploring the information space in detail (Satisfaction Q4: $Mean_{non-BIG}$ = 2.38, $Mean_{BIGexplore}$ = 3.67, $p < 0.001$) and choosing the next target direction (satisfaction Q5: $Mean_{non-BIG}$ = 2.38, $Mean_{BIGexplore}$ = 3.62, $p < 0.001$). In the NASA-TLX (Figure 6), the BIG frameworks showed better results than the *non-BIG* frameworks; however, no significant differences were observed except for mental demand in Experiment 2 (NASA-TLX Q1: $Mean_{BIGexplore}$ = 2.476, $Mean_{non-BIG}$ = 3.143, $p < 0.05$). This was also found in the responses of the participants (P2, P4-P5, P9, P12-P15, P18, and P21). *"Unlike BIG frameworks, non-BIG let me freely explore and investigate the information space without the system loading time."*

Ultimately, from the user's point of view, the impact of *BIGexplore*'s feedback is summarized as follows: 1) It significantly reduces the user's input actions during the information exploration process. 2) Although inconvenient owing to the loading time, it plays a role in reducing the user's mental load when the information space is large. 3) Specifically, the feedback of *BIGexplore* has a significant impact on the user experience in exploration processes with no prespecified targets.

4.2.5 Information Gain Analysis. The uncertainty of the information space decreased when the intended target had a high prior value based on a BIG update. However, because the intended target frequently changed in the changing goal-oriented case, the uncertainty repeatedly increased and decreased. Based on the IG calculation of the BIG framework [19], negative IG was calculated when the uncertainty of the information space increased. To ensure that the BIG framework to effectively reduced uncertainty (= maximizing the IG) during each iteration, the framework was required to detect the point of change of the intended target. To evaluate how the BIG framework decreased the uncertainty in the changing goal-oriented case, we calculated the average IG value of each framework (Table 5). The iterations of *BIGexplore*'s initialization point were excluded when calculating the averaged IG.

Semi-BIGexplore (Experiment 1) and *BIGexplore* (Experiments 2 and 3) showed the highest IG average values. To explain this result more clearly, we illustrate the actual graphs of uncertainty and IG (Figure 11). As shown in Figure 11, in Experiment 1, *semi-BIGexplore* reduces the uncertainty by converging to the answer image. However, in the case of *BIGbase*, the uncertainty does not change significantly during iterations 4–8 owing to a "stuck" status. *Non-BIG* without a specific next view requires more iterations than the other two frameworks because the degree of freedom is high; however, there are many iterations for IG = 0. As mentioned in BIGnav [19], the user action in this iteration is invalid for updating the prior because the system is certain of what users would do. Consequently, to find one correct answer in Experiment 1, *semi-BIGexplore* using the *Sequential_Search* algorithm most effectively reduces the uncertainty of the information space.

In Experiments 2 and 3, the user's intended target changed frequently. Using the *Initialize_Detect* algorithm, *BIGexplore* initialized the information space by maximizing the uncertainty when the target change point was detected. Because of the initialization, *BIGexplore* could avoid a stuck situation and significantly reduce the uncertainty. As shown in Figure 12, since *BIGexplore* initializes whenever the intended target changes, the framework can effectively reduce the uncertainty in other iterations; therefore, it shows the highest average IG value (Table 5). *BIGexplore* also requires the smallest number of iterations with IG < 0, except for the initialization, compared with the other frameworks. However, in the *semi-BIGexplore* and non-BIG graphs, the target change point was not clearly distinguished. The *semi-BIGexplore* graphs in Figure 12 show numerous IG < 0 iterations because *semi-BIGexplore* does not initialize the information space even if the user's intended target changes. In Experiments 2 and 3, the ratio of iterations with IG > 0 among the total iterations for *BIGexplore* averages 88% (*SD* = 7%) and 85% (*SD* = 6%), and for *semi-BIGexplore*, averages 70% (*SD* = 7%) and 66% (*SD* = 6%), respectively. The differences between the *BIGexplore* and *semi-BIGexplore* groups were significant (p < 0.001). The uncertainty graphs of *semi-BIGexplore* exhibit an overall decreasing trend (Figure 12); however, as indicated by the target prediction in Table 3, there is a high possibility that *semi-BIGexplore* applies an incorrect BIG update in the information space. From the IG and uncertainty analysis results, we confirmed that the *Initialize_Detect* algorithm significantly contributes to lowering the uncertainty of the information space.

Table 5: Average IG of all iterations for each framework.

	Information Gain Ranking: *Mean* (*SD*)		
	1st	2nd	3rd
Experiment 1	*semi-BIGexplore*: 0.70 (0.82)	*BIGbase*: 0.32 (0.14)	*non-BIG*: 0.08 (0.04)
Experiment 2	*BIGexplore*: 0.43 (0.08)	*semi-BIGexplore*: 0.20 (0.10)	*non-BIG*: 0.06 (0.02)
Experiment 3	*BIGexplore*: 0.52 (0.10)	*semi-BIGexplore*: 0.18 (0.07)	*non-BIG*: 0.05 (0.02)

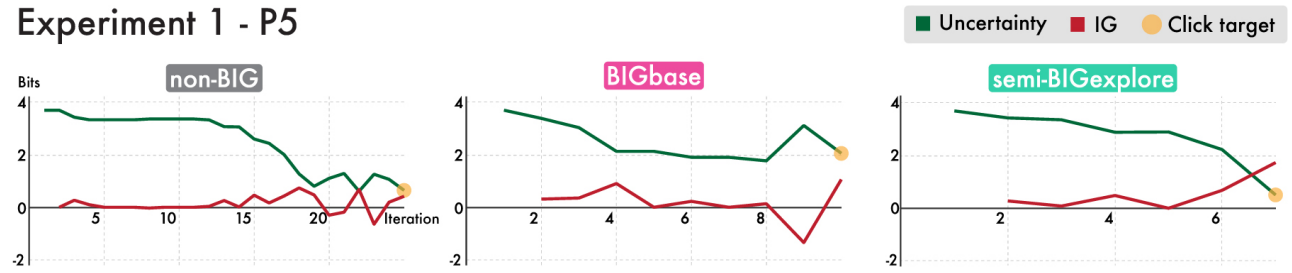

Figure 11: Uncertainty and IG for each iteration in Experiment 1 for P5.

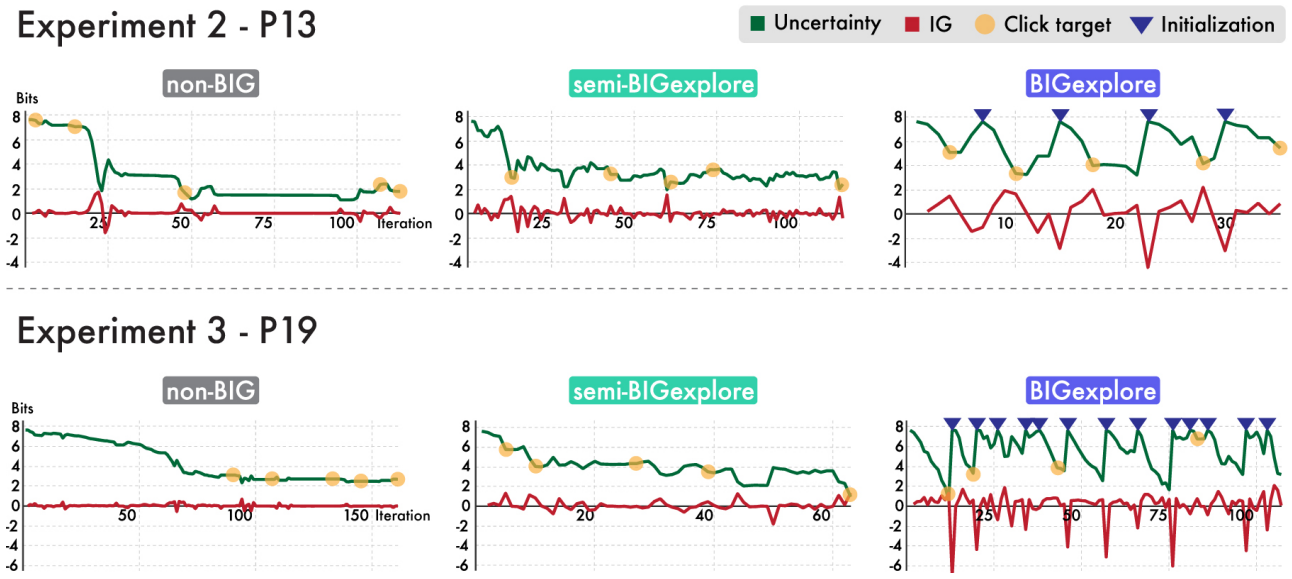

Figure 12: Uncertainty and IG for each iteration in Experiments 2 and 3 for P13 and P19.

4.2.6 BIGexplore for Real World Applications. The *BIGexplore* framework handles changing goal-oriented cases by incorporating the *Sequential_Search* and *Initialize_Detect* algorithms to detect the transition in the browsing behavior of the user. Although we proposed a new vanilla interface for *BIGexplore*, all participants (P1–P21) answered that the image exploration method used in *BIGexplore* was similar to that of the general interface. The participants (P1–P21) also answered that *"Zoom or panning actions are essential for image exploration."* Some of the participants mentioned the absence of a sorting/saving function (P6, P11, P14, P16–P17, and P21) and grid arrangement (P6, P11, P14, P17, and P21) as differences with the general interface. However, these participants also stated that the absence of this function did not affect the search strategy for a given task. They replied that, even with these functions, *"I will zoom in and out of images with zoom actions and explore desired information areas with panning actions."* Thus, the proposed *BIGexplore* can be applied to a general information exploration scenario in various domains as follows:

First, *BIGexplore* can support the basic image exploration process in an image gallery. In addition to the image galleries of smartphones, *BIGexplore* is applicable to all image gallery interface systems. For example, Zhang and Banovis [28] proposed an exploration method that can validate images generated by a generative adversarial network (GAN) in a gallery-type interface. By using

their method, evaluators select photo realistically generated images through the zoom-in, zoom-out, and panning actions on the interface. Because this process belongs to the changing goal-oriented case, *BIGexplore* can be applied to this process to predict the target image of the evaluator. Second, BIGexplore can be implemented as an information exploration support system in Windows File Explorer, as well as for an image exploration. To apply *BIGexplore* to the file explorer system, the current zoom actions can be replaced to moving to the upper/lower folder for the hierarchical information structure. Third, by adding an information similarity model to the user behavior modeling, *BIGexplore* can also support information exploration in an n-dimensional space with the current zoom and panning actions. In this case, the mapped information may be not only be an image data but also clustering or text data. Finally, by probabilistically modeling the zooming actions of the actual camera, *BIGexplore* can predict the target vision of the user in an augmented reality or thermal imaging environment. By appropriately abstracting and modeling for the real world application, *BIGexplore* can effectively support various changing goal-oriented cases.

5 CONCLUSION & FUTURE WORKS

We introduced the *BIGexplore* framework for the information exploration process of changing goal-oriented cases. *BIGexplore* has three distinct advantages over user experiments. First, a significant decrease in the computational cost enables the retrieval process by the user in a large information space. *Semi-BIGexplore* with the *Sequential_Search* algorithm performs better than *BIGbase* in terms of speed and user interaction. Moreover, unlike the *BIGbase* framework, which frequently causes the user to "get stuck" in a view with no images of interest, our proposed *Sequential_Search* algorithm provides a continuous view to the user. Second, the *BIGexplore* framework with the *Initialize_Detect* algorithm detects changes in the user's intended target and initializes the prior for an appropriate probability update. The *BIGexplore* framework showed a significant search performance in changing goal-oriented cases in comparison with other frameworks. Third, from the user's point of view, the feedback of *BIGexplore* helps the user avoid unnecessary commands in the information exploration process and reduces the user's mental load. We validated how the BIG framework helps users in a changing goal-oriented case, along with the contribution of the *BIGexplore* in this study.

To ensure that *BIGexplore* supports more diverse changing goal-oriented cases and maximize the user experience, the following three points should be further studied in the future. The first point is the two hyperparameters α and β. These hyperparameters affect the initialization timing of the information space when the user's intended target changes. Therefore, the α and β should be interactively changed depending on the exploration situation and the user's exploration behavior for more accurate target detection. Second, the user behavior modeling of the BIG framework should include an information similarity model for a better user experience. If the information similarity model is added, *BIGexplore* can be able to provide a next view feedback consisting of images similar to the user's intended target images. Third, a soft initialization method of the information space should be further studied. Currently, the *Initialize_Detect* algorithm initializes the uncertainty of the information space to its maximum value when the user's intended target is changed. However, depending on the exploration situation, even if the intended target is changed, information with a high prior value can be used as feedback in future exploration processes. Therefore, how to initialize the information space and how to use it as feedback in the exploration process should be studied in the future. Although our research was focused on the provisioning of a sequential next view and the detecting the change point of the browsing behavior, *BIGexplore* is expected to support the exploration of the user more efficiently. We expect the *BIGexplore* framework to become a vanilla framework that aids in the information exploration processes in various domains.

ACKNOWLEDGMENTS

This work was supported by a National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIP: Ministry of Science, ICT and Future Planning) (NRF-2020R1C1C1011974).

REFERENCES

[1] John Archea. 1987. *Principles of Computer-aided Design: Computability of Design.* Wiley-Interscience, New York, NY, USA, Chapter 3, 37–52.

[2] Mathieu Aubry, Daniel Maturana, Alexei A. Efros, Bryan C. Russell, and Josef Sivic. 2014. Seeing 3D Chairs: Exemplar Part-based 2D-3D Alignment Using a Large Dataset of CAD Models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* IEEE Computer Society, Los Alamitos, CA, USA, 3762–3769.

[3] Xiaojun Bi, Otmar Hilliges Takeo Igarashi, and Antti Oulasvirta. 2017. Computational Interactivity (Dagstuhl Seminar 17232). *Dagstuhl Reports* 7, 6 (2017), 48–67. https://doi.org/10.4230/DagRep.7.6.48

[4] Sara Bunian, Kai Li, Chaima Jemmali, Casper Harteveld, Yun Fu, Seif El-Nasr, and Magy Seif. 2021. VINS: Visual Search for Mobile User Interface Design. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21).* ACM, New York, NY, USA, 1–14. https://doi.org/10.1145/3411764.3445762

[5] CIPLAB. 2019. *Real and Fake Face Detection.* Retrieved January 14, 2019 from https://www.kaggle.com/ciplab/real-and-fake-face-detection

[6] John J. Dudley, Jason T. Jacques, and Per Ola Kristensson. 2019. Crowdsourcing Interface Feature Design with Bayesian Optimization. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19).* ACM, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300482

[7] Maher Elkhaldi and Robert Woodbury. 2015. Interactive Design Exploration with Alt.Text. *International Journal of Architectural Computing* 13, 2 (2015), 103–122. https://doi.org/10.1260/1478-0771.13.2.103

[8] Stephen Fitchett, Andy Cockburn, and Carl Gutwin. 2013. Improving Navigation-based File Retrieval. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '13).* ACM, New York, NY, USA, 2329–2338. https://doi.org/10.1145/2470654.2481323

[9] Björn Hartmann, Loren Yu, Abel Allison, Yeonsoo Yang, and Scott R. Klemmer. 2008. Design as Exploration: Creating Interface Alternatives through Parallel Authoring and Runtime Tuning. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology (UIST '08).* ACM, New York, NY, USA, 91–100. https://doi.org/10.1145/1449715.1449732

[10] Chathra Hendahewa and Chirag Shah. 2017. Evaluating User Search Trails in Exploratory Search Tasks. *Information Processing and Management* 53, 4 (2017), 905–922. https://doi.org/10.1016/j.ipm.2017.04.001

[11] Forrest Huang, John F. Canny, and Jeffrey Nichols. 2019. Swire: Sketch-based User Interface Retrieval. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19).* ACM, New York, NY, USA, 1–10. https://doi.org/10.1145/3290605.3300334

[12] Kyung Hoon Hyun and Ji-Hyun Lee. 2018. Balancing Homogeneity and Heterogeneity in Design Exploration by Synthesizing Novel Design Alternatives based on Genetic Algorithm and Strategic Styling Decision. *Advanced Engineering Informatics* 38 (2018), 113–128. https://doi.org/10.1016/j.aei.2018.06.005

[13] Florian Kadner, Yannik Keller, and Constantin Rothkopf. 2021. AdaptiFont: Increasing Individuals' Reading Speed with a Generative Font Model and Bayesian Optimization. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21).* ACM, New York, NY, USA, 1–11. https://doi.org/10.1145/3411764.3445140

[14] Mohammad M. Khajah, Brett D. Roads, Robert V. Lindsey, Yun-En Liu, and Michael C. Mozer. 2016. Designing Engaging Games Using Bayesian Optimization. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 5571–5582. https://doi.org/10.1145/2858036.2858253

[15] Yanir Kleiman, Joel Lanir, Dov Danon, Yasmin Felberbaum, and Daniel Cohen-Or. 2015. DynamicMaps: Similarity-based Browsing through a Massive Set of Images. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 995–1004. https://doi.org/10.1145/2702123.2702224

[16] Yuki Koyama, Issei Sato, and Masataka Goto. 2020. Sequential Gallery for Interactive Visual Design Optimization. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 1–12. https://doi.org/10.1145/3386569.3392444

[17] Per Ola Kristensson, Nikola Banovic, Antti Oulasvirta, and John Williamson. 2019. Computational Interaction with Bayesian Methods. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts (CHI'19 Extended Abstracts)*. ACM, New York, NY, USA, 1–6. https://doi.org/10.1145/3290607.3298820

[18] Dan Li and Evangelos Kanoulas. 2018. Bayesian Optimization for Optimizing Retrieval Systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 360–368. https://doi.org/10.1145/3159652.3159665

[19] Wanyu Liu, Rafael Lucas D'Oliveira, Michel Beaudouin-Lafon, and Olivier Rioul. 2017. BIGnav: Bayesian Information Gain for Guiding Multiscale Navigation. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 5869–5880. https://doi.org/10.1145/3025453.3025524

[20] Wanyu Liu, Olivier Rioul, and Michel Beaudouin-Lafon. 2021. Bayesian Information Gain to Design Interaction. In *Bayesian Methods for Interaction Design*, Nikola Banovic, Per Ola Kristensson, Antti Oulasvirta, and John H. Williamson (Eds.). Cambridge University Press, Cambridge, UK. https://hal.telecom-paris.fr/hal-03323514

[21] Wanyu Liu, Olivier Rioul, Joanna McGrenere, Wendy E. Mackay, and Michel Beaudouin-Lafon. 2018. BIGFile: Bayesian Information Gain for Fast File Retrieval. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, 1–13. https://doi.org/10.1145/3173574.3173959

[22] Justin Matejka, Michael Glueck, Erin Bradner, Ali Hashemi, Tovi Grossman, and George Fitzmaurice. 2018. Dream Lens: Exploration and Visualization of Large-Scale Generative Design Datasets. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3173943

[23] Xuehua Shen and ChengXiang Zhai. 2005. Active Feedback in ad hoc Information Retrieval. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '05)*. ACM, New York, NY, USA, 59–66. https://doi.org/10.1145/1076034.1076047

[24] Ehsan Sherkat, Evangelos E. Milios, and Rosane Minghim. 2019. A Visual Analytics Approach for Interactive Document Clustering. *ACM Transactions on Interactive Intelligent Systems* 10, 1 (2019), 1–33. https://doi.org/10.1145/3241380

[25] Kihoon Son, Hwiwon Chun, Sojin Park, and Kyung Hoon Hyun. 2020. C-Space: An Interactive Prototyping Platform for Collaborative Spatial Design Exploration. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. ACM, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376452

[26] Kashyap Todi, Daryl Weir, and Antti Oulasvirta. 2016. Sketchplorer: A mixed-initiative tool for sketching and exploring interactive layout designs. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems (DIS '16)*. ACM, New York, NY, USA, 543––555. https://doi.org/10.1145/2901790.2901817

[27] Robert Woodbury, Arefin Mohiuddin, Mark Cichy, and Volker Mueller. 2017. Interactive Design Galleries: A General Approach to Interacting with Design Alternatives. *Design Studies* 52 (2017), 40–72. https://doi.org/10.1016/j.destud.2017.05.001

[28] Zhang, Enhao and Banovic, Nikola. 2021. Method for Exploring Generative Adversarial Networks (GANs) via Automatically Generated Image Galleries. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. ACM, New York, NY, USA, 1––15. https://doi.org/10.1145/3411764.3445714

A APPENDIX

A.1 Likelihood Modeling

- Case 1

$$P(Y^{(t)} = y^{(t)}|\Theta = \theta^{(true)}, X^{(t)} = x^{(t)}) = \begin{cases} 0 & (y^{(t)} = +) \\ 0.05 & (y^{(t)} = -) \\ 0.05 & (y^{(t)} = \leftrightarrow) \\ 0.9 & (y^{(t)} = *) \end{cases} \quad (1)$$

- Case 2

$$P(Y^{(t)} = y^{(t)}|\Theta = \theta^{(true)}, X^{(t)} = x^{(t)}) = \begin{cases} \frac{k-1}{n} & (y^{(t)} = +) \\ \frac{n+1-k}{3n} & (y^{(t)} = -) \\ \frac{n+1-k}{3n} & (y^{(t)} = \leftrightarrow) \\ \frac{n+1-k}{3n} & (y^{(t)} = *) \end{cases} \quad (2)$$

- Case 3

$$P(Y^{(t)} = y^{(t)}|\Theta = \theta^{(true)}, X^{(t)} = x^{(t)}) = \begin{cases} 0.45 & (y^{(t)} = +) \\ 0.05 & (y^{(t)} = -) \\ 0.05 & (y^{(t)} = \leftrightarrow) \\ 0.45 & (y^{(t)} = *) \end{cases} \quad (3)$$

- Case 4

$$P(Y^{(t)} = y^{(t)}|\Theta = \theta^{(true)}, X^{(t)} = x^{(t)}) = \begin{cases} 0.95 & (y^{(t)} = +) \\ 0 & (y^{(t)} = -) \\ 0 & (y^{(t)} = \leftrightarrow) \\ 0.05 & (y^{(t)} = *) \end{cases} \quad (4)$$

- Case 5

$$P(Y^{(t)} = y^{(t)}|\Theta = \theta^{(true)}, X^{(t)} = x^{(t)}) = \begin{cases} 0 & (y^{(t)} = +) \\ 0.9 & (y^{(t)} = -) \\ 0.05 & (y^{(t)} = \leftrightarrow) \\ 0.05 & (y^{(t)} = *) \end{cases} \quad (5)$$

- Case 6

$$P(Y^{(t)} = y^{(t)}|\Theta = \theta^{(true)}, X^{(t)} = x^{(t)}) = \begin{cases} 0.05 & (y^{(t)} = +) \\ 0.45 & (y^{(t)} = -) \\ 0.45 & (y^{(t)} = \leftrightarrow) \\ 0.05 & (y^{(t)} = *) \end{cases} \quad (6)$$