ANALYSIS OF VIEWING BEHAVIOR OF PROGRAM VISUALIZATION AND INTERACTION WITH INDIVIDUAL DIFFERENCES

by

SHRADHA AMIT KALDATE

(Under the Direction of Eileen Kraemer)

ABSTRACT

Program Visualization (PV) is believed to be useful in Computer Science education. However, while some PVs have been found to help users learn, other PVs have not been beneficial. In this thesis we studied the user's gaze pattern to find effects of popup questions on an individual's visual attention. We further analyzed the correlation of gaze behavior with the individual's preferred learning style, performance based on a pre-test and post-test and a variety of perceptual, attentional and cognitive abilities as determined by a battery of paper-and-pencil and computer-based assessments. While popups appear to be effective for directing attention, no significant effect on comprehension of the depicted algorithm was detected. Individual with different learning styles have distinctive viewing patterns and this finding should be utilized in designing PVs that consider individual differences.

INDEX WORDS: Program Visualization, Eye-Tracking, Individual difference, Popups

ANALYSIS OF VIEWING BEHAVIOR OF PROGRAM VISUALIZATION AND

INTERACTION WITH INDIVIDUAL DIFFERENCES

by

SHRADHA AMIT KALDATE

Bachelor of Engineering, Swami Ramanand Teerth Marathwada University, India, 2002

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment

of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2007

© 2007

Shradha Amit Kaldate

All Rights Reserved

ANALYSIS OF VIEWING BEHAVIOR OF PROGRAM VISUALIZATION AND INTERACTION WITH INDIVIDUAL DIFFERENCES

by

SHRADHA AMIT KALDATE

Major Professor: Eileen Kraemer

Committee: Maria Hybinette
Daniel Everett

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
May 2007

DEDICATION

To my parents and my husband Amit.

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Eileen Kraemer for constant support, guidance and wisdom she has provided throughout my association with her. I would also like to thank Dr. Maria Hybinette and Dr. Daniel Everette for their advice as M.S. thesis committee members. Dr. Beth Davis from Georgia Institute of Technology has provided helpful advice for designing experiments in our study, I am grateful for her support. I would also like to thank my friends and colleagues Manish Agarwal and Phillipa Rhodes for valuable inputs during our discussions.

I am very grateful to my parents, who constantly encouraged and supported me across the globe. My brother is the person who taught me my first computer program and since then he has been constant source of inspiration for me. Needless to say that this thesis would not be possible without my husband Amit. His patience, love, support and much needed advice helped me achieve this.

Thank You to all my friends in Athens whose company made my stay in Athens wonderful and memorable.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Understanding and comprehending algorithms and computer programs is a complex task. Program visualization is a visual representation of the state and behavior of a running program. Due to its graphical properties, program visualization is able to capture the abstract dynamic nature of algorithms not available in the textual code that often describes an algorithm. Program visualization (PV) was believed to be an effective tool for students in understanding algorithms (Lawrence, 1994, Kehoe, 2001, Tudoreanu, 2002). However, even after a long period of development of sophisticated program visualization tools (Brown and Najork, 1998, Stasko, 1998, Helttula, 1990), PV has not been widely used for educational or training purposes. One reason for this could be that studies conducted in this area have shown mixed result (Hundhausen, 2002, Hansen, 2002). Kraemer (2006) lists factors that may contribute to underutilization of PV in the education field. These factors include difficulty and time-consuming effort required in creating the visualization and linking the visualization system to the actual application. From the user's perspective, challenges in navigating the system may be due to poor interaction mechanisms. Lack of trust in the effectiveness of the visualization tool may be a problem for both instructors and users. In this study Kraemer further states that software visualization should be evaluated on its effectiveness and ease of use. In this thesis we study effectiveness of program visualization and factors of the PV that aid in comprehension.

Much of the previous research in the area of program visualization has focused on developing algorithm animations and algorithm animation systems to help users better understand the operation of algorithms (Brown 1987, Reiss 1990, Stasko 1989, Stasko 1990, Stasko 1991, Levy 2000, Cross 2004). These program visualization tools support interactive exploration of graphical representations of programs instead of just animating the algorithm's behavior. Empirical studies have shown that while some program visualizations seem to improve comprehension (Lawerence, 1993, Cosby, 1995, Kann et al., 1997, Hansen, 2000 – Study1, Study2 and Study4) others have failed to demonstrate a benefit (Price, 1990, Gurka, 1996, Mulholland, 1998, Hansen, 2000 – Study3). We seek to understand why some PVs are beneficial while others are not, to identify factors that aid viewer comprehension, and to develop effective program visualizations.

In order to develop these effective program visualizations we first need to know how users perceive and use these graphical representations. Do individuals with different learning styles or capabilities have different ways of comprehending these algorithm animations? Secondly, information about the effect of different cues in a visualization on an individual's attention can be helpful. Designers can use this information to design a visualization that considers the user's attentional, perceptual and cognitive abilities.

In this thesis we apply eye-tracking technology to begin to answer these questions. The eye-tracker is a device that captures a user's gaze pattern as the user views a graphical display while performing some task. When viewing an animation, the point at which users gaze for a period of time is knows as a fixation. We use fixation information to know about the user's attentional process. This will give us the information about which part of the program visualization the user seeks his information from. The change of attention from one part of the program visualization to another is also considered while finding the user's attentional process.

We use the above data from the eye-tracker to find the effects of pop-up questions on the user. To find the effect of popup questions, we divide users into two groups. One group views a visualization of the quicksort algorithm with popup questions in between; another group is shown the same visualization of the quicksort algorithm but without popup questions. While users are viewing the program visualization their gaze pattern is recorded by the eye-tracking device.

We also administer a number of assessments to evaluate a variety of the user's perceptual (ETS I-3), attentional (ETS RL-3) and cognitive (Ekstrom, 1976) abilities and to characterize the user's learning style (Felder, 1988). The user's learning style, capabilities and preferences are determined by conducting a battery of tests which include paper-based and computer based tests. Based on these test results we examine the correlation between learning style and the viewing behavior of users. The results of these examinations will give us insight into whether individuals having a particular learning style view animations in similar ways. This will help us design program visualizations that meet individual needs.

Finally, we administer a pre-test and post-test related to understanding of the quicksort algorithm, in order to evaluate the extent to which viewing the animation resulted in learning.

These analyses will help us in determining a user's perceptual, attentional and comprehension style. This information can be used as guidelines for designing better program visualizations which can be used as a learning aid.

Further, in Chapter 2 we discuss related research work. Chapter 3 defines the methodology for experiments, and in Chapter 4 we discuss the experiment conducted. In Chapter 5 we present our results and discussion and finally, in Chapter 6 we provide conclusions of our study.

Chapter 2

Related Work

This section examines previous research work on program visualization and the evaluation of its effectiveness. We focus on program visualization and its benefits in the field of computer science education. Following this, we highlight some studies that found program visualization to be a useful tool and other studies that did not find it useful. The attempts to analyze this mixed reaction are reported. Further we focus on research that analyzes the cognitive process during the comprehension task. We use eye-tracking methodology to analyze the viewing pattern based on individual differences. Previous work on the use of eye-tracking in the analysis of cognitive processes is reported. Finally, research that focuses on attempts to find individual difference in learning style is reported here.

2.1 Program visualization

Algorithms and data structures are traditionally considered to be difficult to learn (Denning, 1989). Program visualization, which is also referred to as algorithm animation in the computer science field, had shown much promise for helping students understand the dynamic nature of algorithms. Work in this field began in the late 1970's with *Sorting Out Sorting* , an algorithm animation video (Baecker and Sherman, 1981) that visualizes the behavior of several common sorting algorithms. After advances in graphic display technology in the 1980s, there was increased activity in creating sophisticated algorithm animations.

BALSA, an interactive algorithm-animation system that allowed the dynamic execution properties of a program to be visualized, was created at Brown University (Brown and Najork, 1998). It provided a framework through which an instructor could annotate the program code at critical points of program execution. During the execution of the program these annotated critical points would cause a new scene to be drawn that shows a graphical representation of the program at that instance. Sequential execution of annotated points would create animated program behavior displayed on the computer screen. POLKA (Stasko, 1995), a system developed at the Georgia Institute of Technology, supports the animation of concurrent executions.

TANGO (Stasko, 1998) was developed in 1980 and implemented the path-transition paradigm, which supports smooth continuous animations. Aladdin (Helttula, 1990), AlgorithmExplorer (McWhirter, 1996) and Eliot (Lahtinen, 1998) are other systems that let students interact with a program visualization system to build their own animation.

In following years, many other systems with more advanced feature, such as three dimensional graphics (Price, 1998, SW, 1993, BN, 1993) and sound (BH, 1992) were developed. Web-based animation systems were developed, including Collaborative Active Textbooks (Brown, 1997) and Internet Software Visualization Laboratory (Domingue, 1998), which help students to learn algorithms remotely over the network.

2.2 Program Visualization as educational tool

All the program visualization tools mentioned above graphically demonstrate the underlying workings of algorithms. These concepts are believed to be difficult to understand through static methods such as written text and code. One of the prime reasons behind extensive work on developing various kinds of program visualization tools is to use it as a pedagogical tool. Researchers and educators have pursued this intuition of program visualization being a useful pedagogical tool and devoted much time and effort in creating animation systems to help viewers comprehend procedures or concepts of interest. Yet, the use of program visualization for learning algorithms is not widely used in computer science education. One reason for this may be the mixed results reported by empirical studies of effectiveness of program visualization. In the following section we describe these studies.

2.3 Empirical studies for effectiveness of program visualization

Though research in the field of program visualization has been going on for more than a decade, only recently have researchers focused their attention on the effectiveness of program visualization as an educational aid. Empirical studies in this area have shown mixed results.

In the empirical study conducted by Lawrence et al. (1994) three groups of students were evaluated on the basis of their performance on multiple choice questions and free response questions. The three groups varied in the way they were taught about Kruskal's Minimum Spanning Tree algorithm. One group was presented with slides of the algorithm, while two other groups watched an animation of the algorithm in the lab session. Further, the lab session groups were divided into one group that created their own data sets and another group in which students were given files of data sets. The result of this study shows that group who were involved in creating their own data set performed significantly better than the other two groups.

Stasko et al. (1993) presented a study that involved undergraduate students of computer science learning about a pairing heap data structure. The study showed that the group of students who interacted with a textual description along with the algorithm animation performed slightly better than students who interacted only with a traditional textual description. However, no statistically significant result was found.

Hansen et al (2002) developed a system called HalVis. The series of experiments conducted on this system showed a statistically significant result in favor of algorithm animation in study 1, study 2 and study 4.

In study 1 he compared the performance of novice students who used the HalVis system with students who used the textbook chapter, to learn the MergeSort algorithm. The post test scores showed that the HalVis group performed significantly better than the Text group.

In study 2 a similar experiment was conducted, but with advanced students. Students were divided in two groups, one group used HalVis to learn the MergeSort and QuickSort algorithms and the other group used the textbook chapter to learn the same algorithms. From post-test results it was observed that the Halvis group outperformed the Textgroup in this experiment also.

Study 3 compared the effect of HalVis and lecture on novice students. In this study students were divided into two groups: Lecture-Visualization (LV) and Visualization-Lecture (VL) group. The LV group attended a lecture that covered the SelectionSort and MergeSort algorithms first and then interacted with HalVis system. The VL group interacted with HalVis and then attended the lecture. Both groups took the mid-test after their first session either with the HalVis system or in classroom lecture. The post-test was taken after the complete session. The post-test result did not show any significant improvement in performance of either group over the pre-test, whereas, mid-test results were significantly in favor of VL group.

The fourth study compared the effectiveness of HalVis with the Tango algorithm animation system developed by Stasko (1998). One group interacted with HalVis to learn about the Shortest Path algorithm and the other group interacted with the Tango system to learn about the same algorithm. The improvement from pre-test results to post-test results showed that the HalVis group had significant improvement as compared to the Tango group.

Kann et al. (1997) is another study that reports a positive effect of animation on learning. The subject groups were exposed to different degrees of animation such as 1) no animation viewing, 2) viewing of animation, 3) coding the animation of a given task and 4)viewing and coding animation. The problem used was the knapsack problem and student's codes were tested on the concept of recursion used in their code as well as a post-test. The result of studies shows that students who viewed an animation performed better on the post-test than students who did not view an animation. However, no significant correlation was found between students who coded the animation and students who viewed and coded the animation. Further this study reports that student who viewed an animation performed better on procedural and declarative questions; however, the difference was not statistically significant.

Price (1990) conducted an empirical study in which two groups of students were given a task to debug a program. The groups of students varied in the type of debugger, with animation and without animation, used for this task. The result of the study showed that no significant difference was found between the performances of students in the two groups.

Many researchers believed that active interaction of students with the animation system during the learning process can result in a positive effect of program visualization on the learning process. However, studies like Hundhausen and Douglas (2000) do not support this. In this study a group of students who constructed their own visualization did not show any significant difference in accuracy and time required to program and trace an algorithm, from a group who viewed a predefined visualization.

2.4 Analysis of negative results

As empirical studies to evaluate the effectiveness of program visualization as an aid for learning algorithms had produced mixed results , further research to determine the cause of these "markedly mixed" results was carried out. Hundhausen and Douglas (2002) analyzed 24 experimental studies in order to understand the effectiveness of visualization use in learning about algorithms. In their analysis the studies in which there is little difference in effort required by different groups in order to execute a given task, are termed as ‘effort equivalent’ and others in which there is a greater difference in the efforts by different groups is called ‘effort not equivalent’. This study claims that ‘effort not equivalent’ studies are likely to produce more statistically significant result in favor of groups that required more effort.

The study conducted by Saraiya et al. (2004) contradicts the findings by Hundhausen and Douglas (2002). In this study, Saraiya concludes that providing students with sample data sets for important cases in the algorithm increase the pedagogical value of the visualization system significantly. Saraiya further claims that participants who are provided with pseudocode spent more time on the visualization system as compared to other participants who don’t have pseudocode; however, there was no significant difference in procedural understanding by these participants.

2.5 Cognitive process while comprehending algorithm animation

Algorithm animation is the dynamic graphical representation of an algorithm over a given input data set. The goal is to help the user to create a mental mapping with the algorithm process during the visualization. The particular presentation of the graphical features may help or hinder the user in comprehending the algorithm.

The viewing of a program visualization requires various capabilities of the user such as visual and auditory attention (Faraday and Sutcliffe, 1996). Therefore the impact of visualization on a user for understanding an algorithm depends on perceptual, attentional and cognitive properties of both the user and the visualization system. To design effective program visualizations it is important to be aware of the characteristics of both the algorithm animation and of the intended users. However, graphical representations have not been studied enough in terms of their cognitive values (Scaife and Rogers, 1996). They point out that more effort should be made to analyze the graphical representation's role in relation to internal mental representations.

The majority of research in program visualization effectiveness has used means such as improvement in test scores and number of errors made while using the system to evaluate the effectiveness of program visualization. While these methods give an idea about whether a program visualization system helped students in learning the concept or not, it lacks information about how students perceive the visualization. This information is critical because it helps in understanding which aspects of the visualization were able to capture the user's attention and help them in learning and what caused any hindrance in the learning process.

The verbal protocol, sometimes also called the 'think aloud protocol" is widely used in understanding cognitive processes involved with the comprehension of programs or algorithms. In 1986, Letovsky used this protocol to learn about the cognitive process in program comprehension (Letovsky, 1986). In this study users were asked to enhance an existing program. Their activities were videotaped. Participants were instructed to talk as they worked to understand the existing code and to enhance it. The experimenter asked questions of participants such as "what are you thinking?", "where are you looking now?". Data gathered from the verbal protocol was analyzed in fragments to indicate events in the mental process that Letovsky refers to as "cognitively interesting events". In this study he particularly focused on two types of events in the verbal protocol: questions and conjecture. Taxonomies for these events were developed and were analyzed to find a rough theory of the mental model. Since then the verbal protocol has been used by various researchers to find the cognitive process during the execution of the task (Soloway et al., 1988, Littman et al., 1986, Burkhardt et al., 2002, Ko and Uttl, 2003). The purpose of the verbal protocol is to make the cognitive process explicit and draw inferences about the mental state of the participant. The verbal protocol is often combined with other behavior data collected at the same time as direct observations and video (von Mayrhauser, 1996, Ko and Uttl, 2003). However, when it comes to tasks that require high cognitive process and thinking, the verbal protocol may cause a hindrance in the learning process. The use of such a verbal protocol is widely criticized, especially in its use during complex tasks (Branch, 2000, Nielsen et al., 2002).

Researchers have constantly tried to incorporate understanding of the cognitive process in improving the design of the system. Holmquist and Narayanan (2002) have developed a tool that is based on stages that build a mental model to create a visualization. It consist of three different modules namely Hypermedia Authoring Support System (HASS), Hypermedia Education Manual (HEM) and Hypermedia Evaluation System (HES). HASS is an authoring tool that creates HEM, which is a visualization that users interact with during the first phase. A log of user's interaction is recorded. HES analyzes the log data and recommends visualization design improvement. Empirical studies of this tool have confirmed that students who have used the improved version of the visualization outperformed other students.

Later, Narayanan and Hegarty (2002) proposed guidelines for building a presentation based on the cognitive process. They made a presentation of the mechanical working of a flushing system based on these guidelines and compared it to a conventional presentation that did not follow their guidelines. They term the presentation based on their guidelines as a 'cognitive process model based presentation'. The cognitive process model-based presentation has different levels of presentations allowing for comprehension to occur in stages. The empirical studies made three comparisons. Firstly, they compared their presentation with the book that contained the same information. In this comparison, the amount, order and format of the information differed. The second comparison was between the cognitive process model based presentation and a conventional presentation about the same topic. The third comparison differed in the format of presentation only; i.e, the same cognitive process based model presentation on multimedia was compared with a paper-based version of the same presentation. The result of the study showed that in the first two comparisons, the group using the cognitive process model based presentation outperformed the group who used a conventional presentation. However, they did not find any significant difference in the third comparison and thus they claim that the format of information does not matter as long as information is presented based on cognitive process model based guidelines.

2.6 Eye-Tracking

The comprehension of program visualization depends largely on visual stimuli. Therefore the pattern of visual attention during the process of visualization gives an insight into the behavior and strategies user applies in the process of comprehension of visual information. The gaze pattern can also be used to detect whether the user attended to the intended information on the display and thus can be used to improve the design of visualizations. Duchowski (2003) makes the point that even though the location of visual attention is not always a person's point of interest, it provides a strong indication of what a person chooses as her target of interest.

Use of gaze patterns to find a pattern of attention and comprehension has been used since the 1950's. One pioneer work is Fitt et. al (1950) in which they used a mirror and camera mounted on the cockpit to capture the pilot's eye-movement. They conducted frame-by-frame analysis of the pilot's face to get eye-movement data. This study proposes the use of fixation frequency to indicate the importance of each area on the screen. As mentioned in Fitt (1950), “fixation duration can be used as a measure of difficulty of information extraction and interpretation, and the pattern of fixation transitions between displays as a measure of efficiency of arrangement of individual display elements.”

Eye-tracking technology has come a long way since then. With a sophisticated eye-tracking system that track user’s eye movement and analysis software, eye-tracking is now far easier than it was in the 1950’s. However, Sakvucci and Goldberg (2000) point out that there is no standard technique for identifying fixation. Karsh and Breitenbach (1983) also point out that minor changes in the fixation duration definition can result in dramatically different results. Therefore, the measure of the number of fixation durations is often not comparable across two studies.

Eye-tracking is used frequently in usability studies. Kolers used eye-tracking to evaluate the usability of text format on CRT screen (Kolers, 1981). The text of different formats and different scroll rates were presented to 20 participants. Analysis was done by calculating fixation overall, number of fixations of each line of text, and mean fixation duration.

Faraday, Sutcliffe and Alistair used eye-tracking to study comprehension of multimedia presentations and suggested improvements in the design of multimedia presentations (Faraday, 1996). In this study he used an eye-tracking system to record the participant’s eye-movements while they were watching a multimedia presentation named ‘The Etiology of Cancer’. He used a directed graph method in his analysis, in which fixations for each area of interest were presented as arrows from one area of interest to another with the number of fixations marked on the arrow. Constant back and forth of shifts of fixation between areas of interest were interpreted as a difficulty in understanding. He further conducted a recall test and made suggested guidelines for multimedia presentations on the basis of his study.

While there have been many studies using eye-tracking in the field of usability, its use in understanding comprehension of program visualizations or improving their design is very minimal. Crosby and Stelovsky (1990) used eye-tracking to study differences in program comprehension while reading code between expert and novice programmers. Programmers gaze patterns were recorded while they studied Pascal code of the binary search algorithm. Fixation time and number of fixations were used in analysis. Crosby and Stelovsky concluded that expert programmers spent more time on meaningful areas of source code while novice programmer spent more time on comments and comparisons.

Bednarik et al. (2005) used eye-tracking to study gaze behavior of novice and intermediate groups while they comprehended a Java program through a program visualization system called Jeliot3 (Moreno et al, 2004). They used fixation counts for different areas of interest, switches per minute between areas of interests and mean fixation duration during animation for each area of interest in their analysis. They found that intermediate groups mean fixation duration for animation is less than that of a novice group. The attention switches between areas of interest did not show any difference between the novice and intermediate groups.

In a similar study Bednarik et al. (2006) studied visual strategies of low and high program comprehenders. The study found that low comprehenders and high comprehenders have different visual strategies when viewing program visualizations. They observed also that when low comprehenders followed the same visual strategy of high comprehenders their performance was poor. They concluded that eye-tracking can be used to gain insight into the mental model of comprehenders partially.

2.7 Individual Differences

It is known that not all students follow the same strategy when it comes to understanding new concepts or learning new things. This also applies to program comprehension. Nanja and Cook (Nanja, 1987) observed that expert programmers read the program code in the order of execution. Individual difference studies attempt to study the psychological differences between people and their similarities. Since different people have different styles of learning it is important to customize the educational tool to satisfy individual needs. Research with consideration of learning style is lacking in the field of program visualization.

Felder suggested with five scales to determine learning styles of individual (Felder 88). They are sensory – intuitive, visual – auditory, inductive – deductive, active-reflective and sequential – global, corresponding teaching styles are also suggested. Other research that reports learning style models are Curry's learning style model (Curry, 1983) and Kolb's learning style model (Kolb, 1985). There is no agreed-upon learning style model to be used.

Ko and Uttl (2003) studied the effect of individual differences in program comprehension when working with unfamiliar programming environments. In their study, students were given two tasks related to statistics which they had to solve using a commercially available statistical programming package. They performed hierarchical cluster analysis on comprehension behavior. The study concluded that comprehension strategy largely depends on experience. However, debugging performance depended on bug-specific domain knowledge.

While the learning style and individual differences have been widely studied in the field of psychology, its use in the study of effectiveness of program visualization has been minimal.

2.8 Summary

There has been considerable research in development of program visualization systems. Many different program visualization tools with different features have been developed. However, design of these tools was mainly based on the intuition of the designer rather than the student's learning style. Researchers recently have started investigating the effectiveness of program visualization tools as a learning aid. Different methods such as pre-test, post-test scores, comprehension, debugging scores were used to evaluate these systems. Researchers also studied the comprehension strategy of students while they are using these tools. The think-aloud protocol and direct observation are widely used in this process. Recently, use of eye-trackers to detect gaze patterns of users has been used in this field. We study the effectiveness of program visualization using the eye-tracker. Our study differs from the study of Bednarik et al. in the following ways: Firstly, we are investigating if the visual search pattern differs if different cues are presented to students. In our study we are investigating differences in visual pattern when a popup and no-popup version of the same system is presented to students. Further, we are interested in finding out if individual differences play a role in program comprehension strategies.

Chapter 3

Methodology

We believe that visual attention plays an important role in comprehension while viewing program visualizations, and therefore should be studied to promote insight about the patterns users employ to comprehend the algorithms. The purpose of this study is to find if participants who perform better in comprehension of an algorithm, as measured by the difference between pretest and post test results, employ some common visual strategy that is not employed by low-performing participants. Furthermore, we investigate the role individual differences may play in viewing patterns of program visualization. We also investigate if cues in program visualization, such as popup questions, direct user's visual attention to particular parts of an algorithm animation. We use eye-tracking equipment to record the participant's eye-movements on the screen while they watch a visualization of the quicksort algorithm on the SSEA system.

3.1 Eye-Tracking System

An eye-tracker is a device to record eye-movement. Earlier eye-tracking was done by a simple camera recording the user's face and then the user's eye-movements were tracked frame-by-fame (Fitts, 1950). Today there are more sophisticated eye-tracking devices available commercially. Some are used remotely and some are mounted on the participant's head. Most of the eye-tracking devices used today use infrared light emitters and video image analysis of corneal reflections and pupil center to relate them to the direction of gaze. In our experiment we used the 'ASL Eye-Track 6000' eye-tracking system. Figure 3-1 shows the eye-tracking device recording a user's eye-movement while the user is watching the screen. The chin-rest and head clamp shown in the figure are used to keep head movement as low as possible, since large head movement can cause loss of data while recording. The eye-tracking camera will capture an image of the user's eye as he performs his task.

Figure 3.1 ASL Eye-Tracking Device recording user's eye-movement

The setup of the whole eye-tracking system is shown in figure 3-2. The user watches the program visualization on the user screen. The eye-tracker control unit extracts the pupil and reflection of the light source on the cornea from the camera signal and computes pupil diameter and line of gaze. The eye-monitor displays a video image of the eye superimposed with pupil and corneal reflection, which are displayed as a pupil outline and cross-hairs. The scene monitor displays the user screen with the line of gaze superimposed on it. The same output is given to a TV-VCR unit, so that we can record the animation and user line of gaze. The Display PC records all the data which is output from the eye-tracker control unit.

3.2 Eye-Tracker Metric

The Eye-Tracking system provides horizontal and vertical coordinates of eye-position at a sampled rate of time. The sample rate of a typical eye-tracking system is 50 to 240 Hz. At this rate the data gathered by the eye-tracker can quickly add up to a large volume of data to analyze. However, we are not interested in all the data gathered by the eye-tracker. Rather, we are interested in eye-movements of the user known as ‘saccades’ and ‘fixations’.

As mentioned in (Sibert, 2000) we define saccades as rapid movements of the eyes that are executed to reposition the eyes from one location of attention to another one. A single saccade can last between 30 to 120 ms and can span over 1 to 140 degree of visual angle. No information processing is done by the user during saccades.

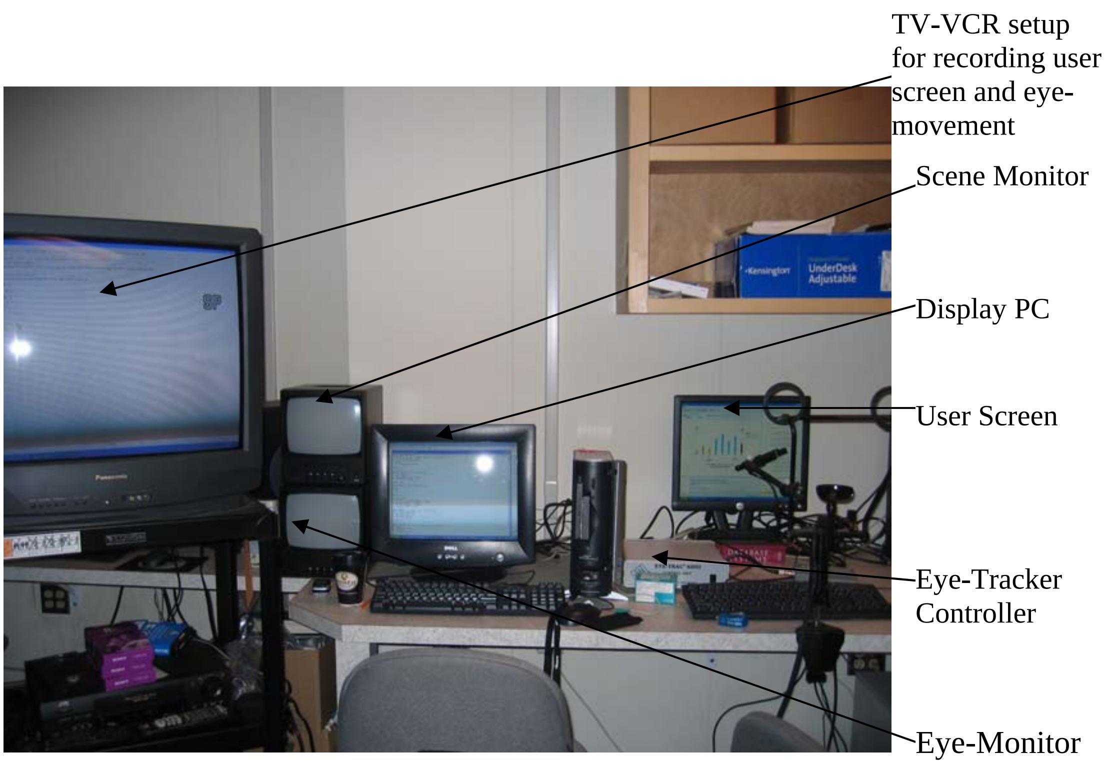

Figure 3.2: ASL-Eye Tracking System setup

Rayner (1998) describes fixations as period when eye movement stabilizes the image of an object on the retina. He further mentions that typical fixation duration is between 200-300 ms.

It is believed that during the fixation the information is processed and therefore these fixation points are considered important when analyzing eye-tracking data.

Areas of interest are the areas on the user's visual display that are of interest to the researcher. The size of each area of interest varies according to the researcher's interest of investigation. Usually the visual area is divided in terms of their function or format of display. In program visualization, typical areas of interest are code area and animation area.

Another metric used to analyze eye-tracker data is called a 'scan path'. The scan path is derived by combining fixation and area of interest. It is an arrangement of a series of fixations over the area of interest. The scan path provides information about the sequence of shifts of visual attention from one area of interest to another.

Eyenal (ASL, 2001) software from ASL was used in our experiments to convert raw data into fixations. Eyenal also provides us with what it calls a fixation sequence, which provides the information about the sequence of fixations and the respective area of interest where the fixation occurred.

In addition to eye-tracking data, a video of the user's gaze pattern gives us as idea of what event was occurring when the user had a particular fixation. In addition to the data provided by the eye-tracker software and video, we added a step number of the animation to the eye-tracking data by synchronizing the time of the experimenters and participant's computer.

3.3 Assessments for individual differences

Researchers are trying hard to improve the design of program visualizations so that the pedagogical importance is increased. It is also known that individuals have different styles of learning, and therefore it is important to know if individual differences in learning style and capabilities have any difference in the way users comprehend a given program visualization.

Thus, the same program visualization can have different impacts on different individuals. This information will help in providing guidelines for designing program visualizations through which designers can meet individual needs and thus improve program visualization's effectiveness. Research in psychology has provided us with various tests to measure individual preferences and learning style and capabilities. Following is brief discussion of the assessments we used in our study.

Felder has created assessments for preferred learning styles (Felder, 1988). His assessments consist of 44 different questions. There are four different dimensions on which individual's preferred learning styles are reported. These scales consist of active/reflective, sensing/intuitive, visual/verbal, and sequential/global. According to Felder, none of the above is the "best" learning style. Rather, it is the preference of the student that can be measured with this scale. He describes these scales as follows.

- An *active* learner prefers learning by doing things such as discussing or applying it. A *reflective* learner prefers learning by thinking about the problem.
- A *sensing* learner likes learning facts and is good at memorizing things and doing hands-on work. An *intuitive* learner prefers discovering things. Intuitive learners are good at grasping new concepts and learn abstract concepts faster than reflective learners.
- *Visual* learners remember best what they see, whereas *verbal* learners prefer text and lectures.
- *Sequential* learners prefer a step-by-step method whereas *global* learners are concerned with understanding a broad overall view of the material.

The ETS surface development test Vz-2 (Ekstrom, 1976) was conducted on participants. This is a cognitive test used to measure spatial visualization ability. Participants are provided with a flat shape with numbered sides and a three-dimensional shape with lettered sides and asked to indicate which numbered side corresponds to which lettered side.

The Figure Classification I-3 by ETS was another test in the assessment series. This test scores the ability to discover rules and explain things. Given two or three groups of figure with three items each, students are asked to determine features that are similar amongst figures within each group and different from other group.

The Inference Test RL-3 by ETS asks student to select one out of five conclusions that can be drawn from each given statement. This test measures fluid intelligence, i.e the ability to reason quickly and to think abstractly.

Working memory refers to the structures and processes used for temporarily storing information. Three computer based memory tests (operating span, reading span and symmetry span) were selected to test working memory of participating students. These tests are available at (http://www.psychology.gatech.edu/renglelab).

The Color Stroop test (Stroop, 1935) measures the effect of interference on performance of a color identification task performed. This test measures distractibility, the effect of interference in the color identification task.

3.4 SSEA System

This study uses SSEA (System to Study Effectiveness of Animations) system (Reed, 2006). The SSEA system provides the program visualization software for the quicksort algorithm. Figure 3-2 shows a snapshot of the SSEA system.

The SSEA system provides the users with the animation and code view of a quicksort algorithm. It allows users to control the speed of animation, go back and forth through any step and to view the animation in step-by-step mode. In animation, different cues such as color, text, labels are used to depict the workings of the quicksort algorithm. The line of code currently being executed is highlighted in the code view. In our study we used two version of SSEA: one with popup and other was one without popup. In the popup version, “popup” questions are asked during the animation. When the question pops-up on the screen the animation is halted and resumes only when user answers the question asked. For more details on the SSEA system refer to (Reed, 2006).

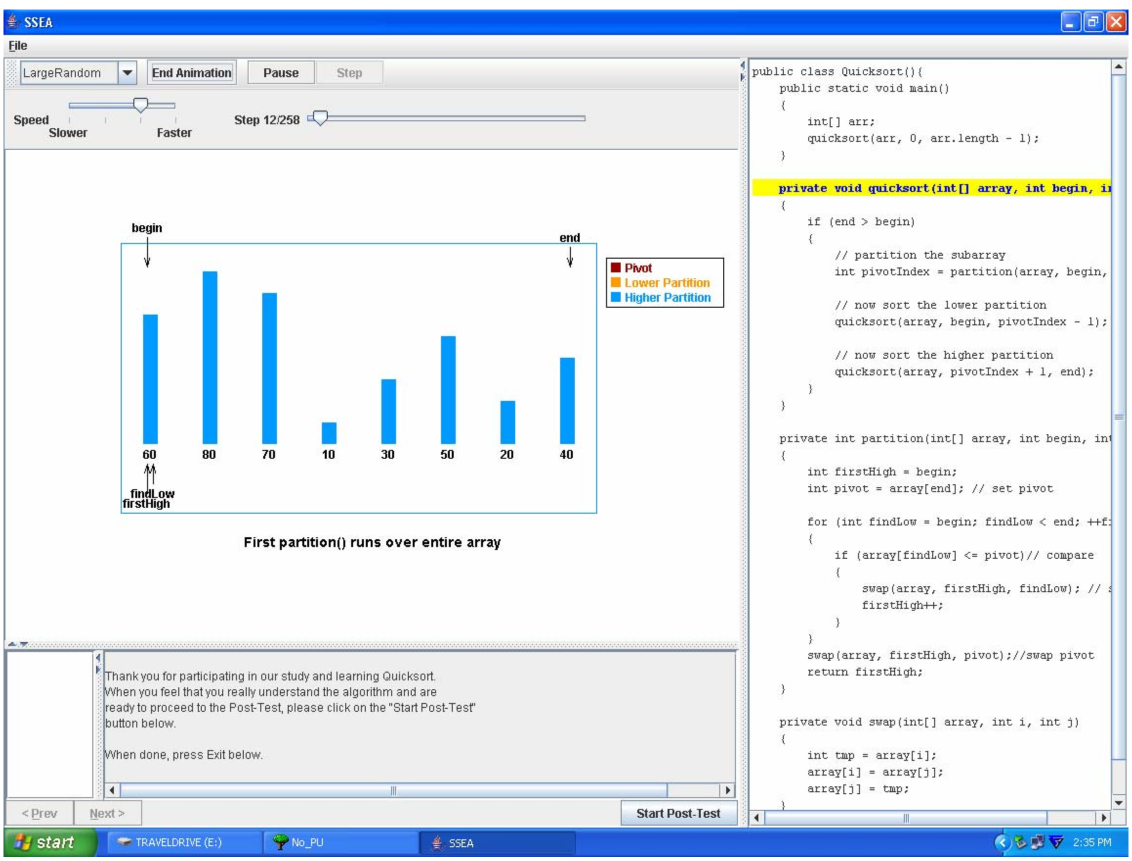

Figure 3.3: Snapshot of SSEA system

3.5 Approach

In this thesis we are investigating user's visual attention by using an eye-tracking system, while they are using a SSEA program visualization. The areas of interest (AOIs) for the SSEA system are shown in figure 3-4. We have divided SSEA into 5 functional areas namely 'controls', 'animation', 'code' , 'instruction' and 'PTestMessage'. Fixation sequence points i.e., fixations and their corresponding AOI's from Eyenal, will help us understand if the user is looking at relevant information while trying to learn about algorithm by viewing these displays. In addition to the fixation sequence, fixation switches are an important part of a viewer's visual information processing. A visual display of fixation points on areas of interest is shown in figure 3-5.

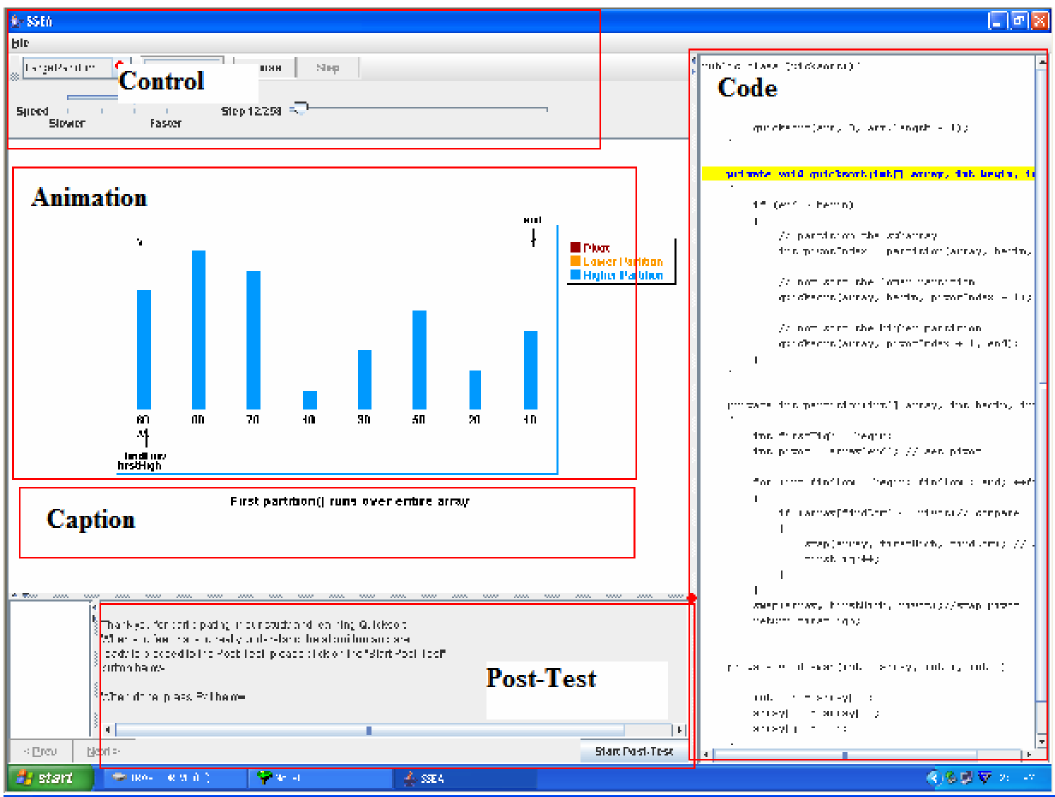

Figure 3.4: Areas of Interest for SSEA system

The popup questions during the animation are "perceptual" questions. These questions mainly focus on animation and ask the user questions such as "which element just swapped?". The list of popup questions is provided in Appendix A. To find the effect of popup questions on participants we examined the user's viewing pattern before and after every popup question and compared it with the behavior of the non-popup group during same period in the animation sequence. To compare the same sequence of animation for the popup and no popup groups we augment the fixation point data with the corresponding step number in the animation when they occur. We then compare fixations for the same sequence by matching step number for popup and no-popup version of SSEA animation.

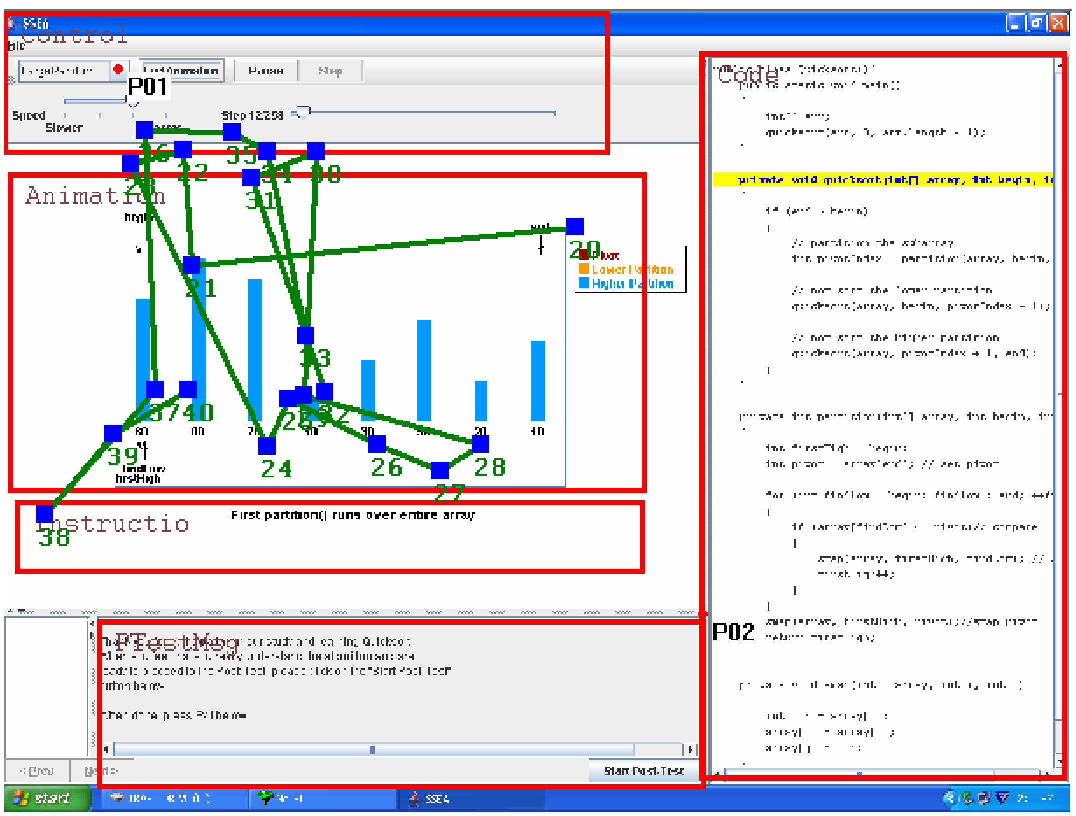

Figure 3.5: Part of Fixation sequence on SSEA System

The pre-test and post-test of traditional questions about the quicksort algorithm are used to score individual's improvement by viewing program visualization. The lists of pre-test and post-test questions are provided in Appendix B. Post-test scores and improvement scores are used to find a correlation with viewing pattern, that is, we will find out if there is any similarity in viewing patterns among high-performing users or low performing users.

The Felder learning style assessments and other assessments give us scores of the participant's ability and preference on different scales. The fixation sequence of participant from each of the Felder's learning style scales are used to find correlation with their preference score of Felder's leaning style assessment. This information will provide us with the information of viewing style preference of the individual of a particular learning style.

Various other tests mentioned above will help in finding further correlation between those test scores and their post-test scores and viewing pattern.

Chapter 4

Experiments

The purpose of this experiment is to analyze the gaze behavior patterns of participants while they interact with program visualizations and to evaluate the differences, if any, between participants who are presented with "popup" questions and those who are not. We further analyze the correlation of gaze behavior with the individual's preferred learning style, performance as measured by a pre-test and post-test, and a variety of perceptual, attentional and cognitive abilities as determined by a battery of paper-and-pencil and computer-based assessments. The fixation duration on each area of interest before and after presentation of a pop-up question will give us insight into whether there is any difference in viewing pattern between a group who were presented with the popup questions and into whether a group who interacted with the no-popup version.

4.1 Participants

Twelve students from the CSCI4800/6800 Human Computer Interaction class of Spring 2007 at the University of Georgia participated in experiments. Students were randomly assigned to either the popup or no-popup version of the program visualization. Out of 12 participants there were 10 male and 2 female students. 4 students were super senior (more than 4 years in school), 7 students were senior undergraduate students and 1 was a junior undergraduate student. All participants had taken the 'Introduction to Computing' class and had knowledge about Java programming, and 5 students reported additional classes in Object Oriented programming. Participants received credit in the Human Computer Interaction class for participation in the experiment. In order to receive the same credit students could write a report describing the design, implementation, and analysis of a study they conducted for this class.

4.2 Materials and Apparatus

The ASL Eye-Trac6000 eye-tracking device from Applied Science Laboratory (ASL, 2001) was used to collect the participant's eye movements. The animation screen with the cross hairs showing users eye movement was video recorded. The SSEA system, described in the previous chapter, was used to visualize the quicksort algorithm. Sorting is a basic concept and is easy to understand, so it is straightforward for participants to understand the stated goal of the program. However, the quicksort algorithm uses a complex divide-and-conquer and partitioning strategy which is more difficult to comprehend.

The various assessment tests listed below were used to gather data about each individual's learning style and ability.

- *Figure Classification Test [ETS I-3]*: Measures the ability to discover rules that explain things. This task requires both concept formation and hypothesis testing.
- *Inference Test [ETS RL-3]*: Measures fluid intelligence, the ability to reason quickly and to think abstractly.
- *Surface Development Test [ETS Vz-3]*: Measures spatial visualization ability.

- *Size Span Test [Cherry 93]*: Measures working memory capacity that requires formation of visual image of an object.
- *Color Stroop Test*: Measures effect of interference in the color identification task.
- *Working memory Test*: Measures the capacity to temporarily store information. Three tests in this group are:
    - Operating Span
    - Reading Span
    - Symmetry Span

The Landolt C test and Ishihara color test were used to check the vision of the participants. Participants who failed these tests were still allowed to go through the experiment, but the data was discarded from analysis. Pre-test, post-test and popup questions used in the test are presented in Appendices A and B.

Since the eye-tracker captures the image of the eye best when there is no bright light, windows in the experiment room were covered with black curtains. Also fluorescent 40 watts, 1900 lumens, 40 watts, 20,000 hours tube lights were installed for Landolt C test.

4.3 Procedure

Participants took part in the experiment over three sessions: an eye-tracking session, a computer-based assessment session, and a paper-based assessment.

Paper-based assessments were conducted during the regularly scheduled class session of the Human Computer Interaction class. Below is the list of test performed during this session.

1. *Figure Classification Test*: There are two parts for this test. Students have 8 minutes to work on each part. The problems on this test have two or three group of figures each consisting of three figures. Students are required to find similarity between the figures that belong in one group and features that are different on figures belonging to different groups. They then allocate eight test figures to the groups they belong. The scores on this test represent the number of figures identified correctly minus fraction penalty for each incorrectly marked figure.
2. *Inference Test*: This test has two parts with 6 minutes to work on each part. In this test students are given one or two statements followed by various conclusions which can be drawn from these statements. Students have to decide which conclusion can be drawn from the statements given. The scores on this test represent the number of correct answers minus fraction penalty for each incorrect answer.
3. *Surface Development Test*: In this test students are presented with picture of irregular, flat shapes, such as pieces of paper or cardboard, and boxes created by folding these shapes. Students are required to find out what lines on the flat shapes correspond to what lines on the boxes. In the flat shapes the dashed lines show you where it can be folded. Students will have 6 minutes for this test. The score on this test is number of correct answer minus fraction penalty for each incorrect answer.
4. *Size Span Test*: In this test students hear names of object and animals. Students are asked to repeat them back in order of their increasing size.

Students get three trials for each series. Students progressively hear longer sequences of words. Full credit is given if the student got series correct in two out of three trials and half credit if only one of three trails was correct. The score for this test is the total of scores of all series.

The eye-tracking session lasted for about 45 mins – 1 hr, depending on the time required for calibration and speed at which the student preferred to view the animation. During this session the Landolt C test and Ishihara color tests were conducted to test vision and color blindness. After these tests we calibrated the eye-tracking device for the participant. For the calibration, the experimenter would adjust the eye-tracker's camera to get a picture of the participant's eye, while the participant looks at the screen with their chin on the chin-rest of the eye-tracking system. Participants are then asked to look at certain points on the calibration screen while the experimenter sets those points for calibration. If the calibration is not correct after several tries, data for that participant is discarded.

After a calibration, the SSEA system is started. Students are initially presented with preliminary demographic questions such as their gender, classes they have taken etc. After that they take a pre-test which assesses their knowledge of the quicksort algorithm. After the pre-test, participants view the animation of the quicksort algorithm. During this period we record the eye-movement. The eye-tracker should be initiated manually to start recording. While watching the animation, participants may change the speed, and go back to a previous step in the animation. Few participants chose to go back and watch the animation once again before proceeding to the post-test. After the animation is finished and the participant has started the post-test, the eye-tracker's recording in stopped. The participant is informed so they may remove their head from the chin-rest and sit more comfortably. The post-test consisted of 16 questions on the quicksort algorithm, 8 of which are similar to pre-test.

The computer-based assessment session lasted for 2:30 hrs, during which students participated in 4 tests, namely: the Color Stroop test, working memory operating span, working memory reading span and working memory symmetry span. These tests are available at http://www.psychology.gatech.edu/renglelab

1. *Color Stroop Test*: There are four sessions in this test. In two sessions students are asked to read the name of the color shown on the screen and in other two sessions they are asked to read color of the text shown on the screen. The experimenter noted the student's response. Colors used in these experiments are red, green, blue, purple and brown.
2. *Operating Span Test*: In this test participants first go through a practice session. In the practice session students are presented with letter for 800ms and asked to recall these letters by clicking on appropriate letters presented on screen. During the practice session students are also presented with simple math problem (e.g. (1*2) +1 = ?). They click the "next" button on screen once they have calculated the answer. On the next screen they have to answer if the number on the screen is the correct answer to the math problem. During this session the student's mean time to solve math problem is calculated and used as limit of time to present a math problem on the actual test. After the practice session student performs the actual test. In the actual O-span test students are presented

first with a math problem and then with letters which appear for 800 ms. Students then have to recall letters as well as the solution to the math problem. The program reports OSPAN score, which is the sum of all perfectly recalled sets.

3. *Reading Span*: This test is similar to the Operating Span test. It has a practice session, during which students are presented with letters on the screen one at a time. Students are asked to recall these letters. For the reading task students are presented with sentences such as “Andy was stopped by the policeman because he crossed the yellow heaven”. Once the participant had read the sentence the next screen is prompt “This sentence makes sense” and students are required to answer true or false. In the actual test the letters and sentences appear one after another and students are required to recall both when they have seen both letters and sentence.

4. *Symmetry Span*: In this experiment students will try to memorize the position of colored squares they see on the screen while they make judgments about other pictures. Students are required to remember where each square was, in the order they are presented. After 2-5 squares have been shown, they will see a grid of 16 possible places the squares could have been, and are required to select each square in the order presented. For the symmetry part, a picture will appear on the screen and students will have to decide if it is symmetrical. After a practice session on each

part, students perform the actual test in which these two tasks will appear interchangeably.

Since these experiments could cause boredom, participants were encouraged to take breaks in between tests.

Chapter 5

Results and Discussion

Out of twelve participants we had to discard three participants' data. One participant's data was discarded because of a low score on the Ishihara color test, another participant's data was discarded because of a calibration failure, and for the third participant we lost the eye-data during the experiment perhaps because of rapid head movement by the participant. We then analyzed the data of 9 participants, 5 of them in the popup group and 4 in the no-popup group.

5.1 Pre-test and Post-test scores

The test-scores of participants on the pre-test, post-test, post-test with similar questions and improvement are given in Table 5.1 for the pop-up group and Table 5.2 for the no-popup group. The questions on the post-test with similar questions are a subset of the post-test in which the questions are similar to the questions on pre-test. The average score of the popup group in the pre-test was 77.50% and the average score of the no-popup group was 68.75%. This shows that the popup group was initially more knowledgeable about the quick-sort algorithm than the no-popup group. The popup group had an average of 82.50% on the post-test with similar questions, and the no-popup group had an average of 84.38%. Improvement from viewing the program visualization showed that the popup group had an improvement of 5.00% whereas the no-popup group had an improvement of 15.63%.

As with prior studies of the effects of popups (Rhodes, 2006) we found a trend towards better performance for the no-popup group, though this trend is not statistically significant. In (Rhodes, 2006) Rhodes suggests that the no-popup group's better performance may be related to the user having the opportunity to see the uninterrupted execution of the algorithm without being required to focus on low-level procedural actions.

Table 5.1: Test Results for Popup Group

Popup Group				
Sub#	**Pre-Test**	**Post-Test**	**Post-Test Similar Question**	**Improvement**
3	62.50%	50.00%	100.00%	37.50%
6	87.50%	87.50%	87.50%	0.00%
7	75.00%	68.75%	75.00%	0.00%
10	87.50%	93.75%	100.00%	12.50%
11	75.00%	50.00%	50.00%	-25.00%

Table 5.2: Test Results for no-popup group

No Popup Group				
Sub #	**Pre-Test**	**Post-Test**	**Post-Test Similar Question**	**Improvement**
2	75.00%	75.00%	100.00%	25.00%
4	37.50%	75.00%	87.50%	50.00%
5	87.50%	75.00%	62.50%	-25.00%
9	75.00%	75.00%	87.50%	12.50%

5.2 Gaze Pattern

We compared the percent of fixation time spent on different areas of interest to find out from which part of the program visualization participants seek information. From five areas of interest namely controls, animation, code, captions and post-test message, only animation, code and caption provide information about the quicksort algorithm. Therefore, we consider these three areas of interest in our analysis.

All participants watched the animation for the longest time followed by caption and then code. Animation had 48.68%, caption had 24.00% and code had 14.33% of fixation time. The remaining 13.00% of fixation time was spent on either controls or reading the post-test message. Figure 5.1 shows the graph for distribution of fixation time between animation, code and captions for all the participants. From this graph we observe that the no-popup group spent more time viewing animation than the popup group, whereas the popup group had more duration of fixation on the code and the caption. The average fixation time spent on the animation is 53.08% for the no-popup group and 45.16 % the popup group, 13.36% on code for the no-popup group and 15.10% for the popup group and 22.44% for the no-popup group and 25.24% for the popup group on the caption. The differences between the popup and no-popup groups in fixation times on animation, code and captions are not statistically significant.

From t-test results on the percent of fixation time on these areas of interests we can say that though we observe a slight variation on viewing pattern of popup and no-popup group these differences are not statistically significant.

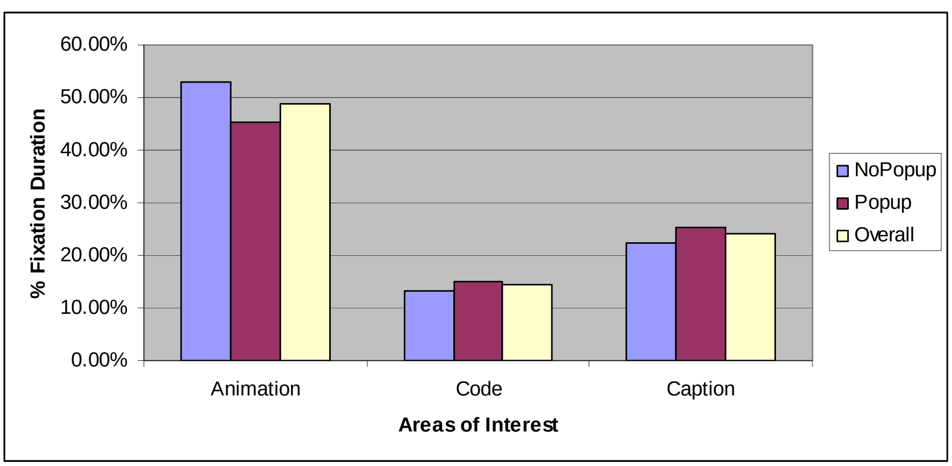

Figure 5.1: Percent of Fixation Time spent on areas of interest by the popup, the no-popup group and overall of all participants

The switching of fixation between areas of interests is another factor in viewing analysis. Both groups had frequent switches between the animation and the caption with the average of 66.67 switches from animation to caption and average of 64.89 from caption to animation. We hypothesize that participants seemed to use the captions when it was difficult to comprehend from the visualization alone. Switches between code and animation followed next in frequency and the fewest switches were between caption and code. There is very less difference in switching behavior between the popup and no-popup groups. Therefore, we can say that popup questions did not have any effect on switching pattern of participants. A barchart of the switching patterns for popup and no-popup groups is shown in figure 5.2.

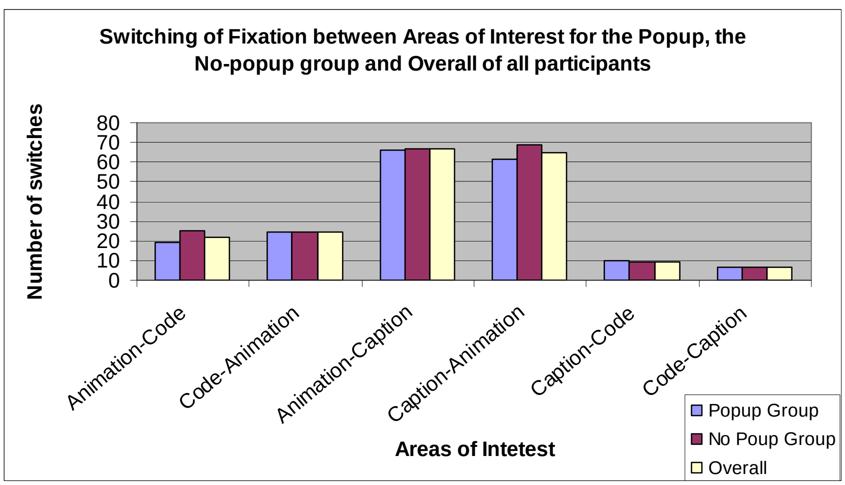

Figure 5.2: Switching of fixation from one area of interest to another for the Popup and the No-popup group

From this we can conclude that participants frequently switched to captions while viewing the animation. This could be because a textual representation of what is happening in animation was helping participants to understand the algorithm. Participants also switched between animation and code. The highlighted part of the code showed the low-level execution of the algorithm. Switching between the code and animation may occur as participants attempt to correlate the animation with the actual execution of the quicksort program. There are very few switches between code and caption. The caption and the code both provided information about animation, by textually describing the process and by highlighting the part of code, but do not directly refer to one another.

5.3 Effects of popup question on gaze pattern

To analyze the effects of popup questions on gaze pattern we compare the fixation duration on each area of interest for the popup group and the no-popup group for the period of animation before and after the popup group attended the popup questions. We also attempt to correlate the type of popup question with viewing pattern. A graph of fixation duration for each area of interest is shown in figure 5.3. We see a sudden increase in the caption fixation duration after popup question 1. After popup 5 there is a sudden increase in fixation duration of code. Popup 3 and 4 did not substantially change the viewing pattern of participants.

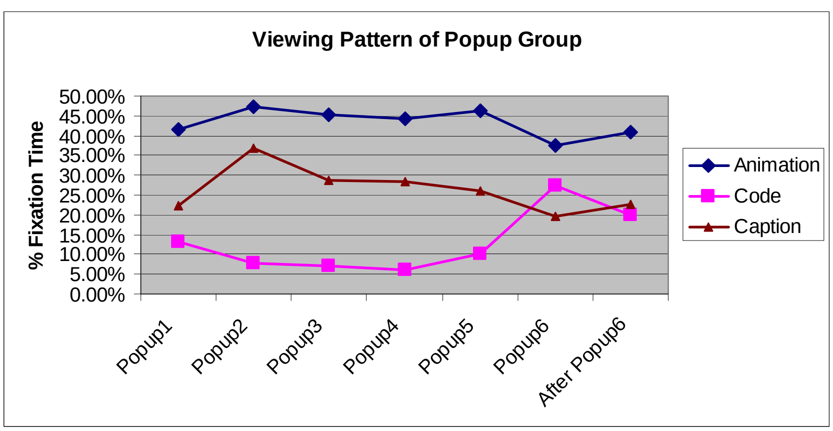

Figure 5.3: Fixation duration before each popup questions

Figures 5.4 – figure 5.6 show the fixation duration for the popup and the no-popup group for each area of interest. From figure 5.4 we see that there is no difference in the pattern of animation viewing until after the last popup appeared. For code, the viewing pattern of the popup group and the no-popup group change drastically after popup 5, (step 183 during algorithm animation), which can be seen in figure 5.5. Figure 5.6 shows that after popup 1 there is a large increase in fixation time on the caption. We believe that popups did not have any effect on the pattern of animation viewing. The drop after the last popup appeared is due an to increase in fixation time on the code.

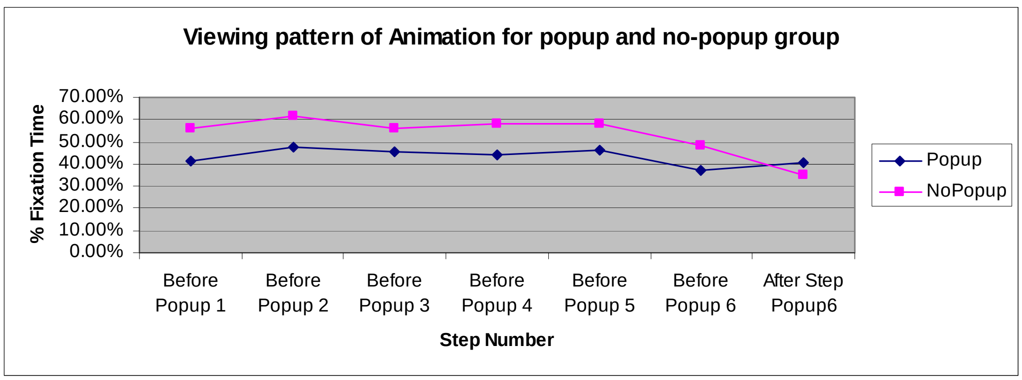

Figure 5.4: Comparison of viewing pattern of popup and no-popup groups for animation after each popup appears for popup group

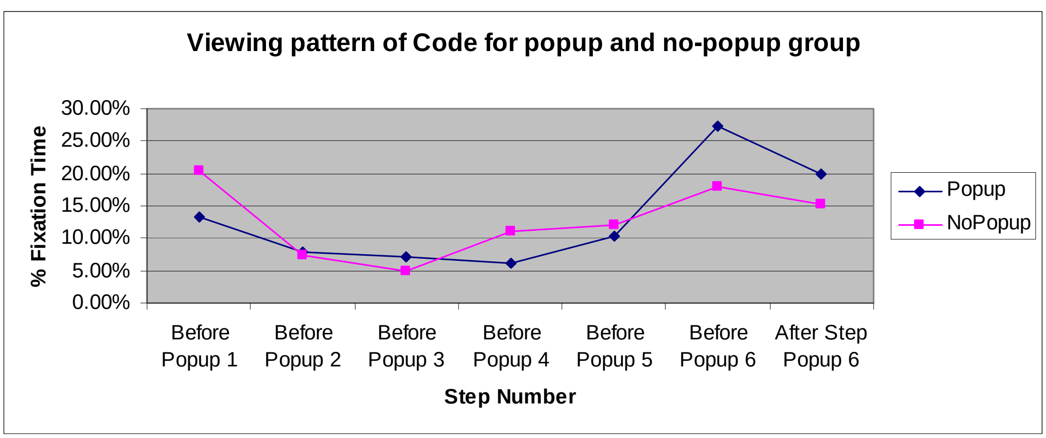

Figure 5.5: Comparison of viewing pattern of popup and no-popup groups for code after each popup appears for popup group

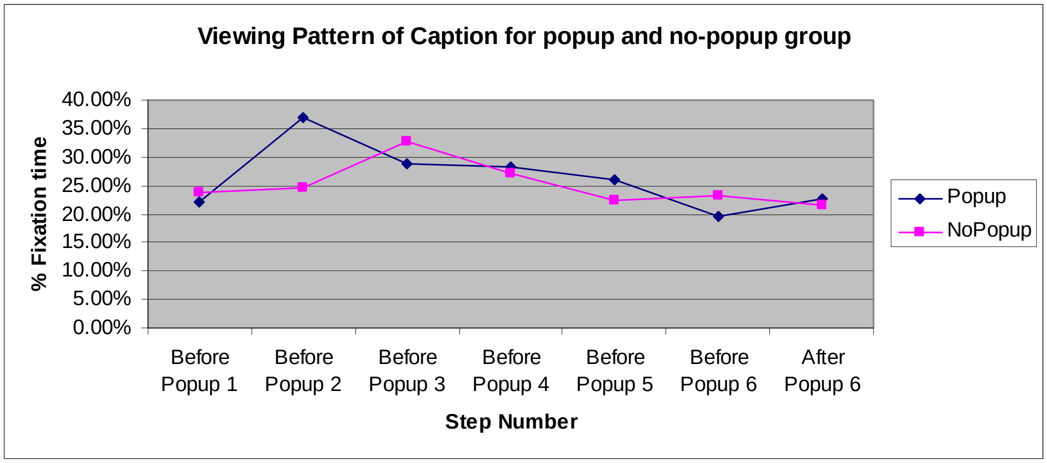

Figure 5.6: Comparison of viewing pattern of popup and no-popup groups for caption after each popup appears for popup group

The sudden increase in caption fixation duration can be attributed to the popup question. The first popup question, which we believe caused an increase in fixation duration at the caption is "Why did 70 and 20 just swap?". The swapping action is shown by the animation but the reason why two elements swapped is provided in the captions. This may be the reason students were paying more attention to captions after popup 1. The question in popup 2 could be answered by looking at animation or code. Increase in fixation duration at code after popup 5 can also be due to the popup question, which was "Which subarray of numbers will be stored next?". This is a predictive question, which asks the user about the next step in the process. There were three predictive questions in the popup question series at popup 3, popup 4 and popup 5. Popup 4 slightly increased in the fixation duration at code, whereas popup 5 caused a large increase in fixation duration at the code. We believe that the predictive nature of the questions caused the participants to look at the code more. The reason could be that by correlating action in the animation with the code, it is easier to predict the next step, whereas watching the animation and the caption gives information about what is happening at that point in the animation.

We also observed that whenever students found the question difficult they changed their viewing pattern. We measure the difficulty level by the percentage of students who answered it correctly. Popup 1 was answered correctly by 60% of participants in the popup group. We saw a sudden increase in fixation duration at the caption. Popup 2 was answered correctly by 40% of students and we saw a drop of fixation duration at caption. Popups 3 and 4 were answered correctly by all students and we see steady pattern during this period. Popup 5, which caused a sudden increase in fixation at the caption was answered correctly by 60% of participants.

We conclude that the predictive nature and difficulty level of the popup questions prompted users to change their viewing pattern.

5.4 Learning Style

As mentioned in the previous chapter the Felder learning style test reports a result on 4 scales. Scores of participants on learning styles are shown in tables below Table 5.3 through Table 5.5, below.

Active-Reflective scale: Participants belonging to the active and reflective groups are shown in Table 5.3. Four participants were active learners and five were reflective learners. The active learners group had equal number of participants who participated in popup and no-popup experiment. In the active group two participants were high active learners, one of whom belongs to the popup group and the other belongs to the no-popup group.

Three out of five reflective learners participated in the popup experiment and two participated in the no-popup experiment. One participant had a high reflective score and one participant had a low reflective score and both participated in the popup experiment, thus balancing the high reflective score. Three moderate scoring participants were mixed in the popup and no-popup groups.

Table 5.3: Active and Reflective Group

Active Group			Reflective Group	
Group	*Score*		*Group*	*Score*
Popup	7		Popup	1
Popup	5		Popup	7
NoPopup	9		Popup	3
NoPopup	3		No-Popup	3
			No-Popup	3

Sensing and intuitive scale: Seven participants were sensing learners and two were intuitive learners. Five out of seven participants in the sensing group participated in the popup group and three participated in the no-popup group. All participants with low scores in sensing learning style were in the popup group and moderate scoring participants were in the no-popup experiment.

The intuitive group had two participants, evenly distributed to the popup and no-popup groups. Neither participant had a high score of intuitive learning style. Table 5.4 shows participants belonging to sensing and intuitive group and their corresponding scores.

Table 5.4: Sensing and Intuitive Group

Sensing Group			Intuitive Group	
Group	*Score*		*Group*	*Score*
Popup	3		Popup	1
Popup	1		No-Popup	3
Popup	3			
Popup	1			
No-Popup	5			
No-Popup	5			
No-Popup	5			

Sequential and Global scale: Six participants had a sequential learning style and three participants preferred a global learning style. Four out of six participants in the sequential learning style group belonged to the popup group and two belonged to the no-popup group. One participant was a high sequential learner and belonged to the popup group.

The global group had three participants: two were in the no-popup group and one was in the popup group. All three participants in this group had low to moderate scores. Table 5.5 shows participant belonging to sequential and global groups.

All the participants were visual learners and their scores ranged from moderate to high.

Table 5.5: Sequential and Global Group

Sequential Group			**Global Group**	
Group	*Score*		*Group*	*Score*
Popup	3		Popup	3
Popup	1		No-Popup	3
Popup	9		No-Popup	1
Popup	1			
No-Popup	1			
No-Popup	1			

5.5 Correlation of learning style with gaze pattern

A correlation coefficient describes the degree of relationship between two variables. A positive value of correlation coefficient means higher scores on one variable tend to be paired with higher scores on the other and that lower scores on one variable tend to be paired with lower scores on the other. A negative value of correlation coefficient means higher scores on one variable tend to be paired with lower score on the other and vice-versa. The correlation of two variables *x* and *y* is given by

$$r = \frac{n\sum xy - \left(\sum x\right)\left(\sum y\right)}{\sqrt{n\left(\sum x^2\right) - \left(\sum x\right)^2}\sqrt{n\left(\sum y^2\right) - \left(\sum y\right)^2}}$$

where n = number of pairs of data.

Using Felder’s learning style questionnaire we group students into two groups for each scale as described in the section above. We found that participants in two groups on these scales have a distinctive gaze patterns. We could not use the sensing-intuitive scale or the visual-verbal scale because the former scale had too few participants in intuitive group to compute a correlation and the latter scale did not have any participants in the verbal group.

Table 5.6 shows the correlation of both groups with fixation duration on each area of interest. For active learners we found a moderate correlation with fixation duration of animation (0.4739) and for reflective group we did not find any significant correlation with animation fixation time (-0.17). Correlation with code was moderate for both the active and reflective groups. Both groups showed high correlation with fixation duration on captions, with the reflective group having a high positive correlation with fixation time on captions (0.82).

Though the correlation for animation was low for the reflective group we can observe that it had negative correlation (-0.17) with fixation duration at animation, where as the active group had a positive correlation (0.47). The same pattern can be found on fixation duration for code and captions. Figure 5.7 shows correlation for active and reflective group.

Table 5.6: Correlation score for active and reflective group with fixation duration

% of Fixation Duration	**Active**	**Reflective**
Animation	0.47	-0.17
Code	0.55	-0.6
Captions	-0.94	0.82

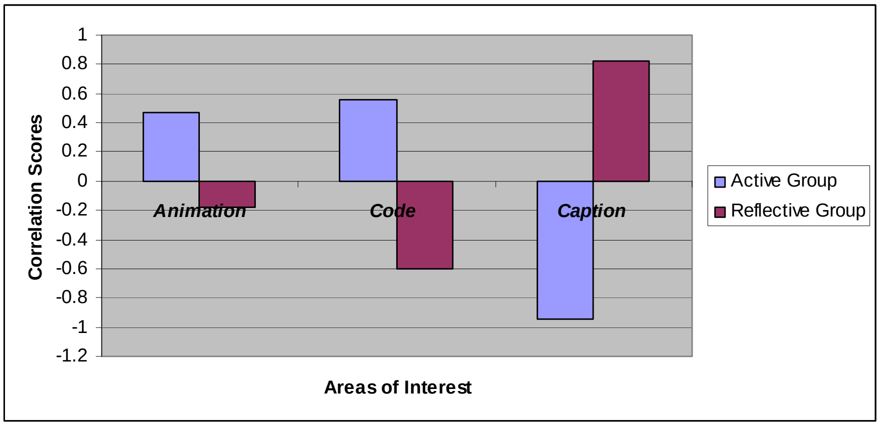

Figure 5.7: Correlation of Active and reflective scores with percentage of fixation at various areas of interest

From this correlation data we can conclude that highly active learners spend more time reading code and less time reading captions, whereas highly reflective learners will spend less time reading code or looking at the animation and focus more on captions. Active learners are more likely to spend time on animation as compared to reflective learners. This can be explained by the definition of active and reflective learners. Active learners tend to learn more by applying or discussing things. In this case we can say that by spending more time on code and animation rather than reading captions, active learners might be applying or correlating high-level concepts with the code. Reflective learners like to think more before applying those concepts, therefore we observe reflective learners showing a positive trend towards captions which help them in clarifying concepts, rather than focusing on code that provides low-level details of the animation.

Table 5.7 shows the correlation for the sequential and global groups of learning style and gaze behavior. The sequential group has a moderate negative correlation with fixation duration for animation (-0.535), whereas the global group has a high positive correlation (0.858) with same. Both groups show very high correlation with code and moderate to low correlation with fixation duration for captions, but in opposite direction. Figure 5.8 shows the correlation of global and sequential scores with the duration of fixation at each area of interest.

We found distinctive patterns of viewing for the sequential and global groups. The sequential group showed a negative correlation with fixation duration of animation and the global group showed a high positive correlation for it. Similarly, a high positive correlation is found for sequential group with code (0.800) and moderate negative correlation for fixation duration of captions (-0.456). Distinct opposite viewing patterns in terms of correlation can be found for the global group, with a high negative correlations for code-viewing (-0.818) and a moderate positive correlation for caption viewing (0.365).

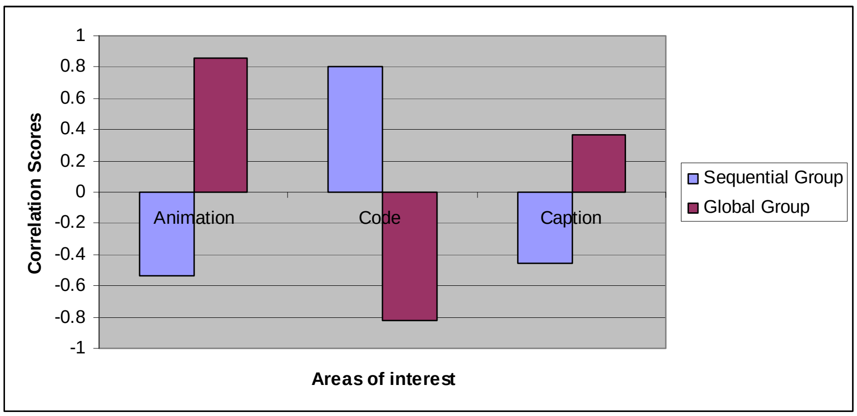

Figure 5.8: Correlation of Sequential and Global scores with percentage of fixation at various areas of interest

Table 5.7: Correlation score for sequential and global group with fixation duration

% of Fixation Duration	**Sequential**	**Global**
Animation	-0.535	0.858
Code	0.801	-0.818
Caption	-0.456	0.365

This result is as expected for the global and sequential groups. Sequential learners tend to learn by following linear steps, each following logically from another, which is the nature of code and captions. Global learners tend to learn more by looking at random patterns and then putting things together, therefore we observe highly global learners tend to look more at animation rather than at code.

5.6 Correlation of paper based test with test scores

The correlations between the various paper based assessments and the post-test scores and pre-test to post-test improvement were calculated. We observed correlations for some of these tests while for others there were no correlations or the reason for a strong correlation could not be explained. The correlation of these tests could not be found for the post test for the no-popup group because all the students in this group scored equally well and thus there was no variation in their performance.

5.6.1 Inference Test

The correlation for the Inference Test (ETS RL-3) with the post-test for both groups is shown in table 5.8. The inference test has a higher correlation with the post-test scores for both groups considered together. The posttest score for all sixteen questions showed a moderate positive correlation with the inference test score (0.61). The inference test score also has moderate positive correlation with the eight the post test question similar to the pre-test questions (0.64) and improvement from pre-test to post-test (0.54).

As mentioned in chapter 4, the inference test measures the ability to reason quickly and to think abstractly. These skills seem to help students in comprehending the algorithm. Figure 5.9 shows a barchart of the correlation of inference test scores with the post-test scores for the popup group, the no-popup group, and overall.

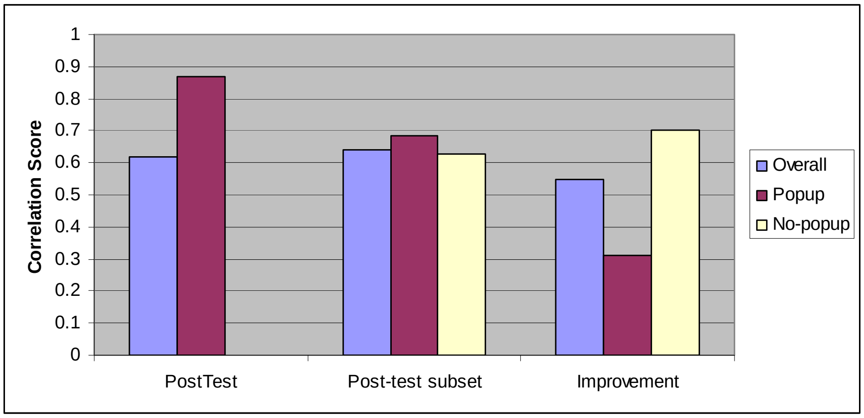

Figure 5.9: Correlation of Post-test scores with Inference test scores

Table 5.8: Correlation of Test scores with Inference Test

Overall	**Correlation with RL-3**
Posttest	0.619580318
Posttest with similar questions	0.641641846
Improvement	0.547041378
Popup Group	
Posttest	0.866240682
Posttest with similar questions	0.684897407
Improvement	0.312956803
No Popup Group	
Posttest	
Posttest with similar questions	0.626166802
Improvement	0.701493424

The correlation of the inference test with gaze pattern is shown in Figure 5.10. There is very low correlation for fixation time for animation (0.057). However, fixation duration for code has moderate negative correlation (-0.49) and fixation duration for caption has moderate positive correlation (0.497). It shows that students with high fluid intelligence tend to look less at code and more at the caption. Table 5.9 provides these correlation scores.

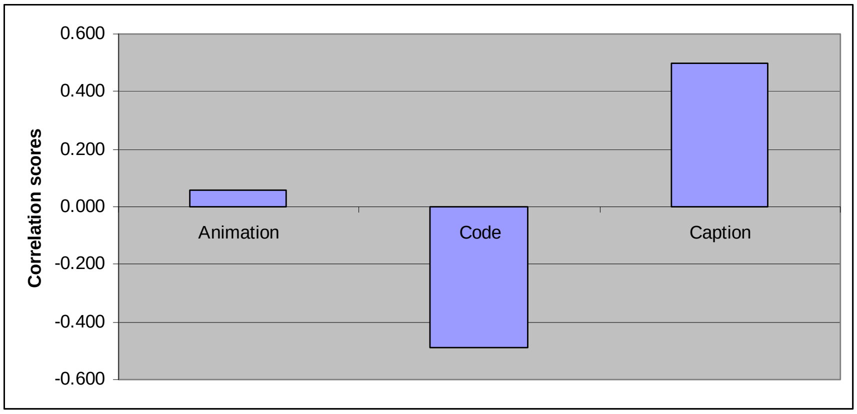

Figure 5.10: Correlation of Inference Test Scores with duration of fixation at each area of interest

Table 5.9: Correlation of Fixation duration with Inference Test

Overall	RL-3
Fixation Time for Animation	0.057913457
Fixation Time for Code	-0.490078225
Fixation Time for Instruction	0.497873572

5.6.2 Figure Classification Test

The correlation of the figure classification test with post-test scores and improvement, shown in figure 5.11, is not clear. The figure classification test shows a negative correlation on the post-test with similar questions and on improvement, overall and for both groups separately. The no-popup group showed a high negative correlation with the figure classification test for the post-test with similar questions (-0.97) and for improvement from pre-test to post-test (-0.82). Table 5.10 shows correlation scores for figure classification test with post-test scores and improvement from pre-test to post-test for each group.

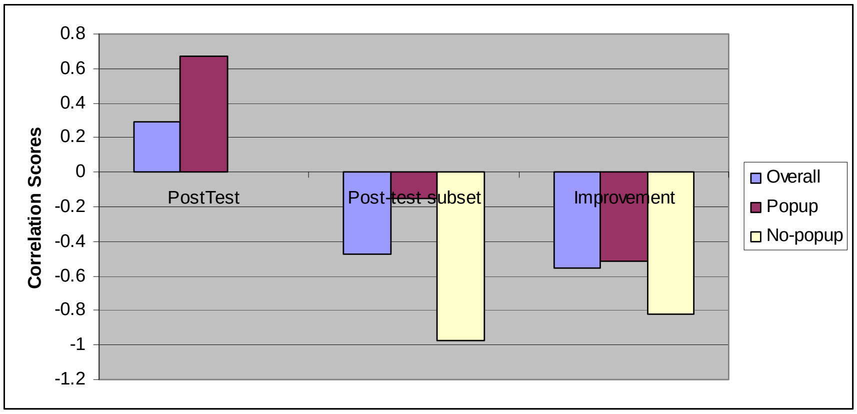

Figure 5.11: Correlation of figure classification scores with post-test scores

Table 5.10: Correlation of test scores with figure classification test scores

Overall	**Fig Class I-3**
Post-Test	0.295107197
Post-Test with similar questions	-0.478080235
Improvement	-0.553917794
Popup Group	
Post-Test	0.673357132
Post-Test with similar questions	-0.150435483
Improvement	-0.510980538
No Popup Group	
Post-Test	NA
Post-Test with similar questions	-0.977003427
Improvement	-0.824400401

This result contradicts our assumption that better performance in the figure classification test should result in better performance on the post-test and improvement. The reason for this negative trend is not clear. No significant correlation was found for the figure classification test and fixation pattern.

5.6.3 Surface Development Test

Overall, the surface development test has a low negative correlation with the post-test (-0.11), a low positive correlation with the post-test with similar questions (0.06) and with improvement (0.302). The surface development test shows a low negative correlation (-0.19) for the popup group and a high positive correlation (0.89) for the no-popup group on the post-test with similar questions. Improvement from pre-test to post-test scores also have a low negative correlation (-0.08) for the popup group and a high positive correlation (0.937) for the no-popup group. As the surface development test measures spatial visualization ability of the individual, we would imagine that such abilities should help students in comprehending the algorithm through visualization. The negative correlation of popup group may be due to distraction because of popup questions in process of understanding the algorithm. Since the no-popup group did not have this distraction we could observe higher positive correlation with the surface development test. Another explanation is that the popup questions helped to direct the attention of viewers in a way that overcame weaker capabilities for spatial visualization. Figure 5.12 shows correlation of the surface development test scores with the post-test scores and table 5.11 provides values of these correlation scores.

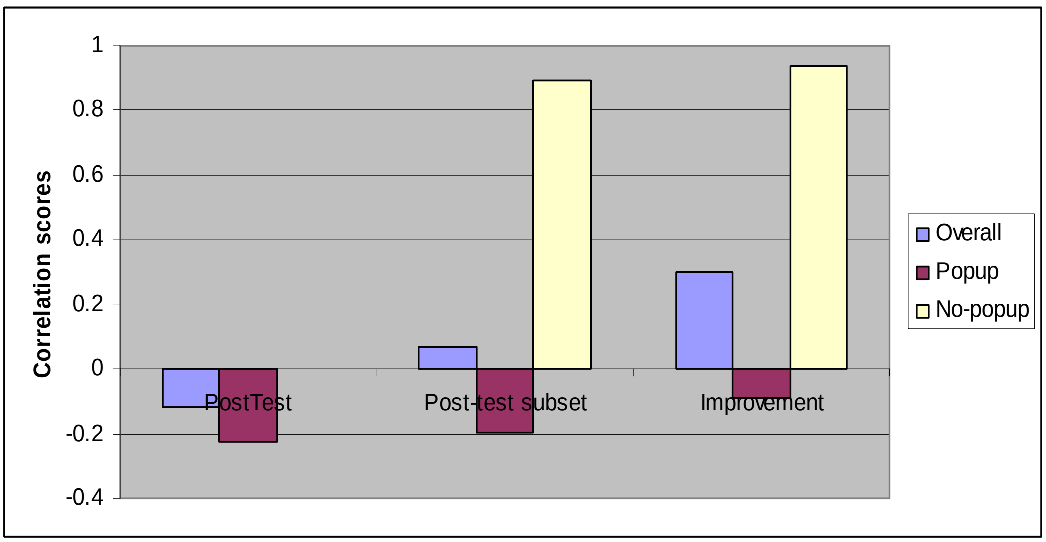

Figure 5.12: Correlation of Surface development test scores with post-test scores

Table 5.11: Correlation of surface development test scores with post-test scores

Overall	**Surface Deve Vz-3**
Post-Test	-0.118003359
Post-Test with similar questions	0.066128079
Improvement	0.302136323
Popup	
Post-Test	-0.224367318
Post-Test with similar questions	-0.198215215
Improvement	-0.089348859
No Popup	
Post-Test	
Post-Test with similar questions	0.892471612
Improvement	0.937593743

5.6.4 Size Span Test

The size span test showed a low positive correlation with the post-test scores (0.33) whereas moderate negative correlation on post-test with similar questions (-0.37) and improvement for both groups (-0.45). The no-popup group had high negative correlation on subset of post test similar to pre-test (-0.927) and on improvement (-0.866) where as the popup group had low negative correlation on post-test with similar question (-0.159) and improvement (-0.294). This negative correlation is against the intuition of size span test which measures the working memory capacity that requires formation of visual image of an object. So, if the participant had high score on size span test he can visualize more images or objects on screen. It could be thought of as keeping more objects in the memory will not help in comprehension. However, the reason for the high negative correlation for the no-popup group and low negative correlation for the popup group is unclear. Figure 5.13 shows a barchart for correlation scores of size span test and post-test. Table 5.12 provides values for these correlations.

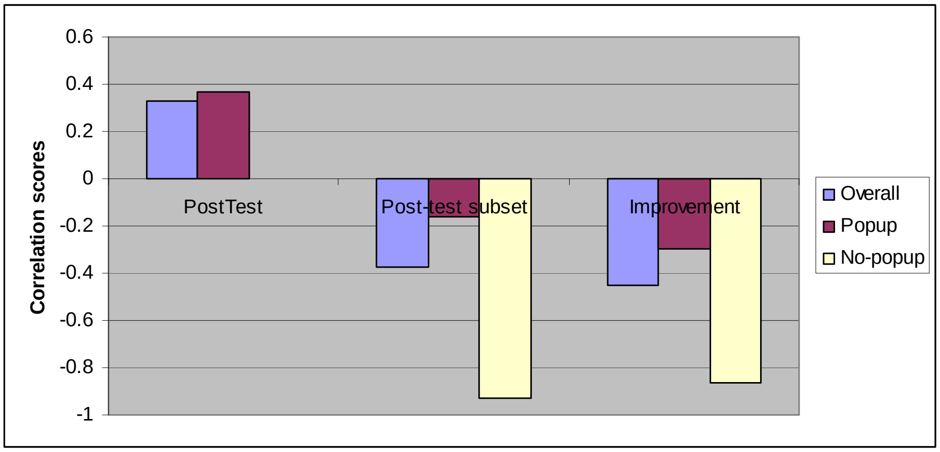

Figure 5.13: Correlation of size span test scores with post-test scores

Table 5.12: Correlation of size span test scores with post-test scores

Overall	**Size Span Test**
Post-Test	0.331290374
Post-Test with similar questions	-0.373101254
Improvement	-0.453927937
Popup	
Post-Test	0.367668656
Post-Test with similar questions	-0.159719141
Improvement	-0.294244943
No Popup	
Post-Test	NA
Post-Test with similar questions	-0.92717265
Improvement	-0.866666667

5.7 Correlation of computer based test with test scores

5.7.1 Reading span test

The reading span test had moderate positive correlation with the post-test scores of both groups for all participants as shown in figure 5.14. The popup group had moderate correlation with reading span on subset of posttest (0.589) and improvement (0.433) whereas the no-popup group had a high positive correlation for the post-test with similar questions (0.809) and a moderate correlation (0.618) for improvement. Reading span had a very low negative correlation with fixation duration on animation and code. However, it has a low positive correlation (0.321) for fixation duration on the caption. Table 5.13 shows the correlation of the reading span test with post-test and improvement from pre-test to post-test.

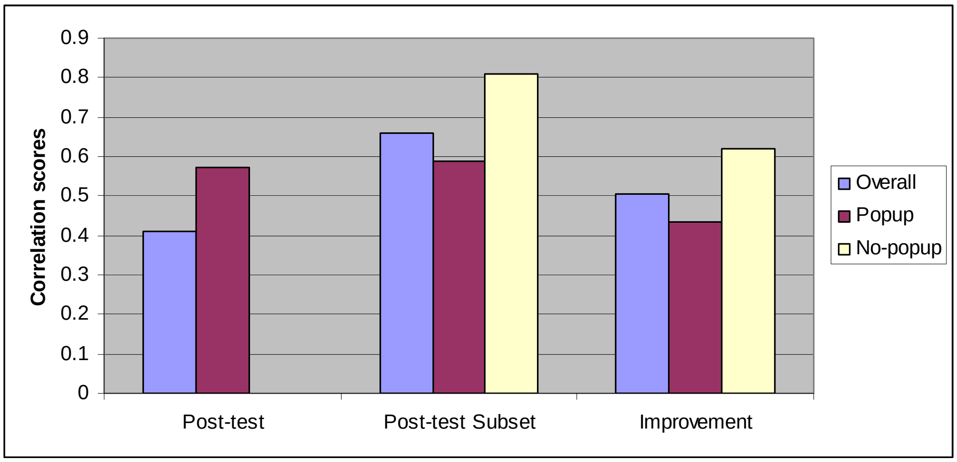

Figure 5.14: Correlation of Reading span test scores and post-test scores

Table 5.13: Correlation of Reading span test scores and post-test scores

Overall	**R-Span**
Post-Test	0.410306
Post-Test with similar questions	0.659394
Improvement	0.505046
Popup	
Post-Test	0.571594
Post-Test with similar questions	0.589319
Improvement	0.43337
No Popup	
Post-Test	NA
Post-Test with similar questions	0.809175
Improvement	0.61807

Reading span measures working memory for reading and comprehending sentences, so the positive correlation with fixation on caption is understandable. We can also say that students with high reading span performed better on the post-test and fixation on captions may have helped these students with high reading span to read textual information and keep it in working memory which helped them in the post-test.

5.7.2 Operating Span Test

The operating span test that measures working memory capacity for operations has a moderate positive correlation with the post-test scores of all participants. The popup group showed a high positive correlation in the subset of the post-test similar to the pre-test (0.764) and improvement from the pre-test to the post-test (0.701). The no-popup group showed moderate negative correlation for the post-test with similar questions (-0.598) and improvement (-0.618). Figure 5.15 shows the trend for this correlation and table 5.14 provides correlation scores for the same.

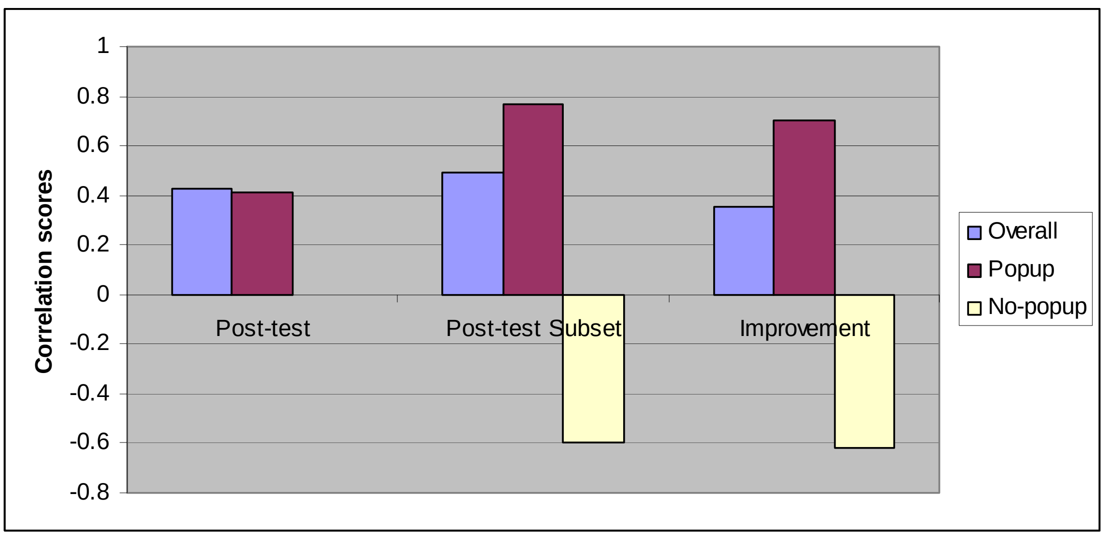

Figure 5.15: Correlation of Operating Span test scores with post-test scores

Table 5.14: Correlation of Operating span test scores with post-test scores

Overall	O-span
Post-Test	0.42812
Post-Test with similar questions	0.493097
Improvement	0.351948
Popup	
Post-Test	0.412958
Post-Test with similar questions	0.764913
Improvement	0.701144
No Popup	
Post-Test	NA
Post-Test with similar questions	-0.59874
Improvement	-0.61858

This correlation shows that student with high a operating spans performed better when they were presented with program visualization with popup questions. The reason for this could be that popup questions helped students with high a operating span to keep those operations required to answer the popup question in memory which helped them in post-test. Only those popup group members with a high operating span were able to perform well despite the interruptions. The reason the negative correlation of operating span with post-test score for the no-popup group is not clear.

5.7.2 Symmetry Span Test

The symmetry span test showed a moderate correlation with the post-test scores (0.476) and a low positive correlation on improvement (0.269). The no-popup group showed a high positive correlation with improvement (0.831) and low positive correlation of with the subset of the post-test (0.26). The result shows that student with higher symmetry span tend to achieve greater comprehension of the visualized algorithm. Figure 5.16 and table 5.15 show correlation for symmetry span scores with post-test scores and improvement.

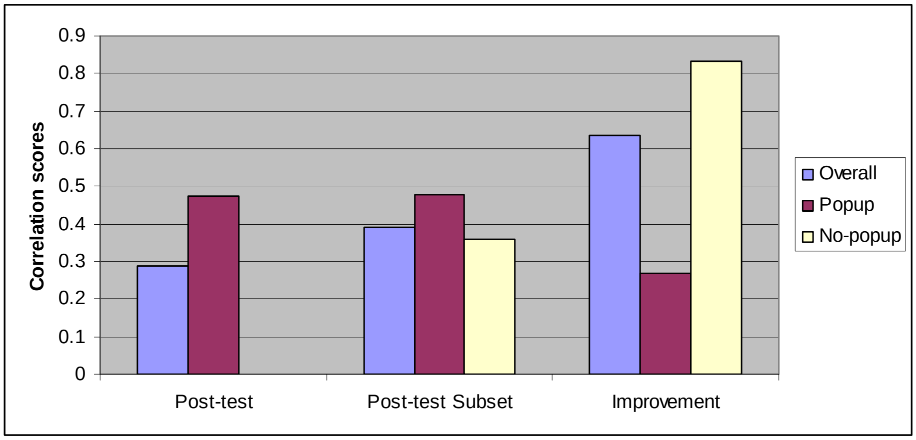

Figure 5.16: Correlation of symmetry span scores with post-test scores

Table 5.15: Correlation of symmetry span scores with post-test scores

Overall	**S-span**
Post-Test	0.287705
Post-Test with similar questions	0.39064
Improvement	0.633858
Popup	
Post-Test	0.474424
Post-Test with similar questions	0.476112
Improvement	0.269533
No Popup	
Post-Test	NA
Post-Test with similar questions	0.359048
Improvement	0.831297

5.7.2 Color Stroop Test

The Color Stroop test, which measures distractibility, showed a low positive correlation for the popup group (0.195) whereas the no-popup group showed a high positive correlation (0.89) on subset of post-test. The trend of correlation is shown in figure 5.17. The reason for this difference in correlation between the color distraction measure and the test scores for popup and no-popup groups is not clear. Table 5.16 shows scores of these correlations.

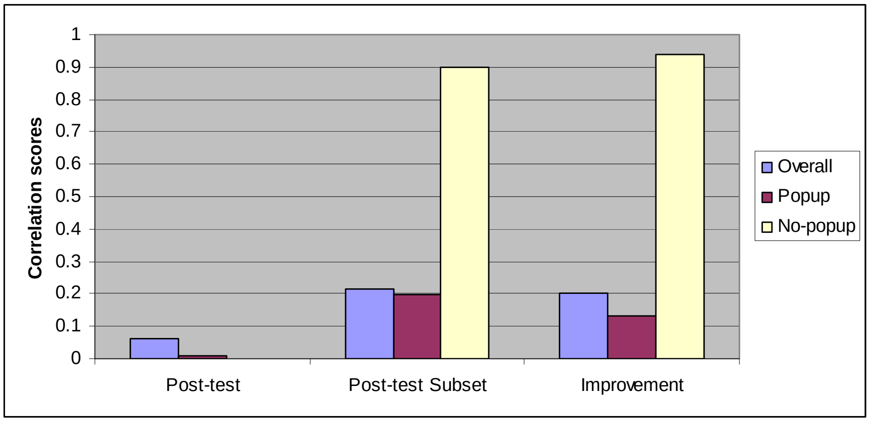

Figure 5.17: Correlation of Color Stroop test scores with post-test scores

Table 5.16: Correlation of Color Stroop test scores with post-test scores

Overall	**Color Stroop**
Post-Test	0.05963
Post-Test with similar questions	0.216295
Improvement	0.202254
Popup	
Post-Test	0.010371
Post-Test with similar questions	0.195982
Improvement	0.130725
No Popup	
Post-Test	NA
Post-Test with similar questions	0.899229
Improvement	0.940019

The working memory span tests showed positive correlation with the post-test scores, except for the no-popup group on operating span. We can say in general that higher working memory span will help students perform better on tests of comprehension.

5.8 Correlation of VBMG clustering with viewing behavior

We group the participants into separate groups based on their viewing behavior. Groups are formed using “Viewing Behavior Model Graph” (VBMGs) Markov models of user gaze behavior. Details on the clustering based on VBMGs refer to [Agarwal 07].

The VBMG clustering formed three clusters. Cluster 1 had five participants, three of whom interacted with the popup version of SSEA and two who participated with the no-popup version of SSEA. Cluster 2 had only one participant, in the no-popup group. Cluster 3 had three participants two of whom were in the popup group and one who was in the no-popup group.

We calculated the correlation of the distance of participants with the centroid of the cluster they belong to with various assessments scores and viewing behavior to see if participants belonging to the same cluster show similar behavior on various tests scores and viewing behavior.

To assign a numeric value to each node in the cluster, we use the formula (1 – distance-from-centroid). In this way, a high score (close to 1) is assigned to nodes close to the centroid and thus most representative of the cluster and a lower score is assigned to nodes further from the centroid and this less representative of the cluster.

The correlation of the clusterscore and fixation duration at various areas of interest is shown in table 5.17. From table 5.17 we can see that participants belonging to cluster 1 has moderate positive correlation with fixation duration at code (0.422), whereas participants belonging to cluster 3 had moderate negative correlation with fixation duration at code (-0.671). Cluster 1 has high negative correlation with fixation duration at caption (-0.73) and cluster 2 has high positive correlation with fixation duration at caption (0.982).

The participants in different clusters shows distinct viewing pattern in terms of code and captions. Participants in both the clusters have very low negative correlation with fixation at animation. Therefore we can say that there is not much of a difference in viewing pattern of animation for participants in both clusters. However, participants closely belonging to cluster 1 (closer to centriod of cluster 1) are likely to seek more information from code rather than from caption and participants closely belonging to cluster 2 are likely to seek more information from caption rather than from code.

Table 5.17: Correlation of VBMG cluster distances with fixation duration at areas of interest.

	Animation	Code	Caption
Cluster 1	-0.10742	0.422209	-0.73609
Cluster 3	-0.08874	-0.67166	0.982997

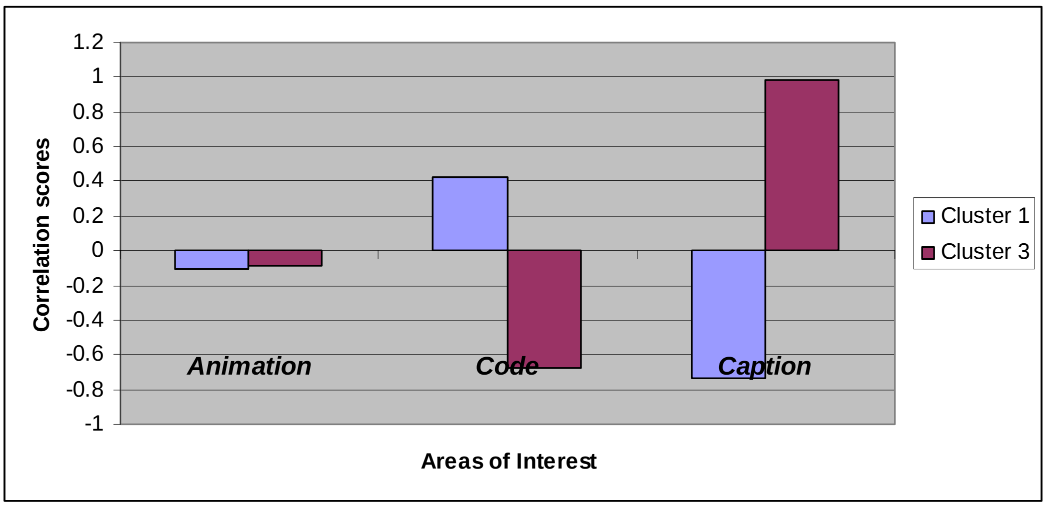

Figure 5.18: Correlation of Cluster distance with fixation duration

5.9 Correlation of VBMG clustering with paper based assessments

The correlation of clusterscore with paper based assessments will give us insight on similarity of cognitive, attentional and perceptual capabilities of participants belonging to same cluster. Table 5.18 shows the correlation of distance from cluster centroid with scores on various paper based assessments. We see that cluster 1 has very low positive correlation for inference test (0.035) however cluster 3 shows very high positive correlation with this test (0.971). On figure classification test participants in cluster 1 shows low positive correlation (0.143) and participants in cluster 3 shows moderate negative correlation (-0.554). Cluster 1 has moderate positive correlation with surface development test (0.445) and cluster 3 has high negative correlation (-0.80). Size span test shows very low correlation for cluster 1 however, cluster 3 shows moderate negative correlation (-0.592).

Table 5.18: Correlation of paper-based test and cluster distance

	Inference Test - RL 3	Fig ClassI-3	Surface Deve Vz-3	Size Span Test
Cluster 1	0.035458822	0.143751145	0.44527259	0.008915988
Cluster 3	0.971901132	-0.554819482	-0.802034802	-0.59206198

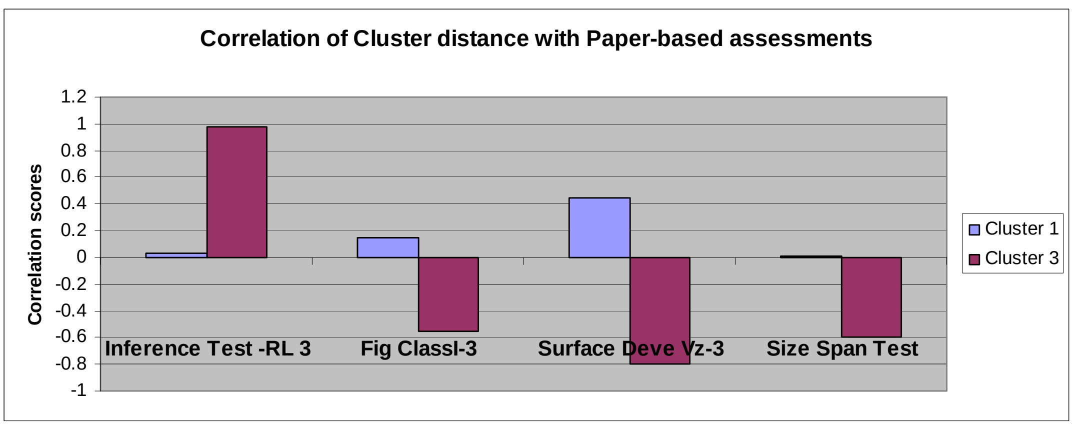

Figure 5.19: Correlation of cluster distance with scores on various paper based assessments

From the figure 5.19 we see the trend that participants closer to cluster 3 are good at fluid intelligence but they show negative trend for figure classification, surface development and size span test. These users tended to seek information from the captions rather than code. Participants in cluster 1 did not have significant correlation for inference test and size span test, but they show a positive trend figure classification and surface development abilities. These users tended to seek information from the code rather than from captions.

5.10 Correlation of VBMG clustering with computer based assessments

Correlation of clusterscores and scores on computer based assessments are provided in table 5.19. Cluster 1 shows low negative correlation (-0.20) and cluster 2 shows high positive correlation (0.91) with the reading span. Scores for symmetry span has low correlation for cluster 1 (0.30) and moderate correlation for cluster 3 (0.516). Cluster 1 (-0.664) and cluster 2 (-0.487) showed moderate negative correlation with operating span. Both group had low positive correlation on the Color Stroop test, cluster 3 (0.345) with slightly higher correlation than cluster 1(0.162).

Table 5.19: Correlation of cluster distance with computer based assessments

	R-Span	S- Span	O-Span	Color Stroop
Cluster 1	-0.20646	0.301437	-0.66419	0.1624415
Cluster 3	0.919095	0.516455	-0.48709	0.3453204

Figure 5.20 shows graph for correlation of cluster distance with the computer based assessments.

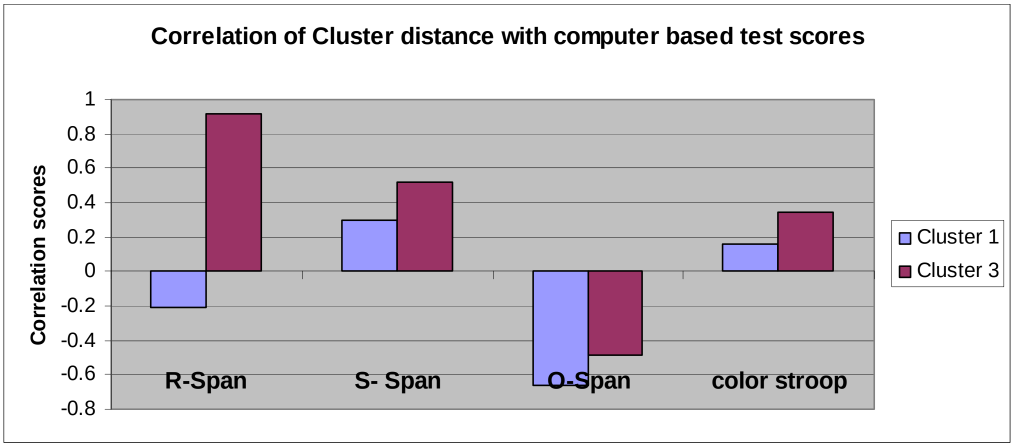

Figure 5.20: Correlation of cluster distance with computer based assessments

Cluster 1 and cluster 3 exhibit different patterns on reading span. Participants closer to the centroid of cluster 3 are likely to have better reading span. Both clusters shows a similar pattern on the symmetry span, the operating span and the Color Stroop test. Participants closer to centroid are likely to have better symmetry span and better able to handle interference due to color, whereas they are likely to have low score on the operating span test.

The clustering was based on the transition of fixation from one area of interest to another. The correlation of cluster distance with the various assessments shows whether participants with similar viewing pattern have other similarities. We observe that participants in cluster 3 have better fluid intelligence than participants in cluster 1, where as participants in cluster 1 have better visualization capability. The clusters did not show any difference in working memory capacity except for reading span. Participants in cluster 3 are likely to have better reading span than participants in cluster 1 and they also spent more time reading captions.

5.11 Correlation of VBMG clustering with post-test scores.

The correlation of clusterscores and post.test scores are provided in table 5.20. Cluster 1 shows low negative correlation on post-test (-0.128) and cluster 2 shows moderate positive correlation (0.483) with the post-test. On the subset of post-test with questions similar to the pre-test participants in cluster 1 showed a low negative correlation (-0.250) and participants in cluster 3 showed a high positive correlation. Participants in both the clusters have high positive correlation, cluster 3 showing higher correlation of (0.938) than cluster 1 (0.667). Figure 5.21 shows a barchart for these correlations.

Table 5.20: Correlation of cluster distance with post-test scores and improvement

	Post-test	Post-test Subset	Improvement
Cluster 1	-0.128965276	-0.250365993	0.667140135
Cluster 3	0.483362594	0.773752196	0.938484852

From figure 5.22 we see that participants who are closer to cluster 3 tend to comprehend better from the program visualization. However, the average score of participants in both clusters shows that participants in cluster 1 had better performance in post-test and had better improvement from post-test. Table 5.21 shows average scores of participants in these cluster for the pre-test, the post-test, the subset of post-test similar to pre-test and the improvement from pre-test to post-test. We see that participants belonging to cluster 1 tend to show better improvement than participants in cluster 3.

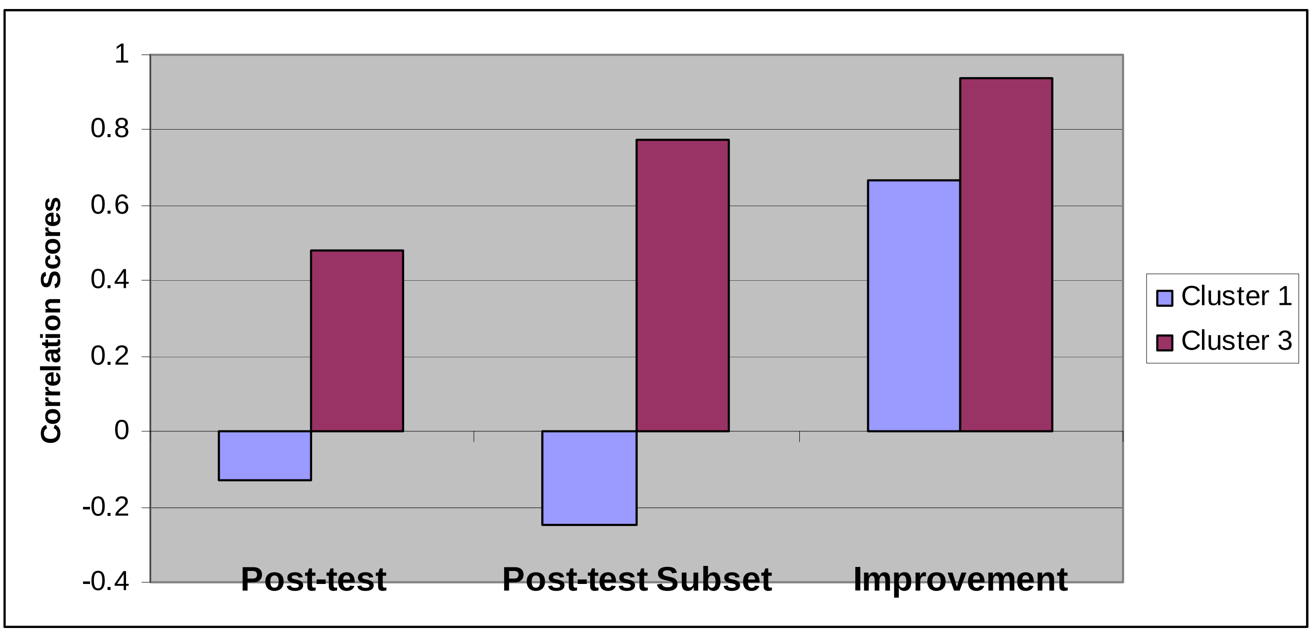

Figure 5.21: Correlation of cluster distance with post-test scores

Table 5.21: Average scores of pre-test, post-test and improvement for both clusters

	Pre-test	Post-test	Post-test Subset	Improvement
Cluster 1	67.50%	72.50%	92.50%	25.00%
Cluster 3	83.33%	70.83%	66.67%	-16.67%

Chapter 6

Conclusions

The work presented in this thesis is an attempt to analyze effective features of program visualization based on individual differences. We have conducted an empirical study to find gaze patterns of participants while they learned the quicksort algorithm using the SSEA program visualization. A battery of paper-and-pencil-based assessments and computer-based-assessments gave us information about the individual's learning style, capabilities and preferences.

The pre-test and post-test score difference showed that the no-popup group comprehended better than the popup group, though the result was not statistically significant. Prior work (Rhodes 2006) involving popup and no-popup question showed a similar trend. As mentioned in the study by Rhodes, we also believe that the reason for better performance by the no-popup group may be due to the lack of distraction in their natural viewing and understanding process, as opposed to the popup group participants who were interrupted by these popup questions.

The study of gaze pattern for the popup group and the no-popup group shows that the popup group spent more time on code and captions while the no-popup group relied more on the animation for their information. We believe that popup questions increase fixation on captions and code since the information in those popup questions was available from that part of algorithm animation. However, we note that these differences in viewing pattern are not statistically significant.

The switching of fixation from one area of interest to another did not differ much between the two groups. However, study of these switching patterns showed us that all participants had a high number of switches between animation and caption, followed by animation and code. There are very few switches of fixation between caption and code. We conclude that it is helpful for students to have a textual description of the animation, since students try to seek information from the text when the visualization is not clear.

We analyzed the effect of popup questions on viewing pattern. We did not observe any correlation between viewing pattern and correct answer for that question. Therefore we can conclude that participant use their prior knowledge, and understanding of the algorithm from the animation to answer those questions. We however conclude that popup questions do guide a participant's attention to a particular part of animation. In our experiment we see that a popup question whose information was presented in a caption before the popup, caused a sudden rise in fixation duration at caption. Similarly, predictive questions caused participants to look at code. We also observed the pattern that whenever participants found questions difficult, as measured by average score of participants on that popup question, there was change in fixation duration for these areas of interest.

From the viewing analysis and post-test scores we can conclude following

- Popups during animation can cause distraction in their natural understanding process.
- Popups are successful in shifting attention of users, so if necessary they can be used to direct the user's attention to focus on important parts of the animation.
- We also recommend further study with large number of participants to obtain statistically significant results.

The learning style of individuals showed a trend that can be used in designing algorithm animations with the individual's needs and preferences in mind. Our study demonstrated that highly active learners focused more on animation and code whereas reflective learners preferred reading captions. Sequential learners read code more than global learners, who focused on animation and caption. We could not find a correlation for the visual-verbal group or sensing-intuitive group because of lack of enough students in one of the groups.

We could use this information in designing the algorithm animation. We believe that all three forms of information (animation, code and caption) should be presented on the screen. However, while designing animations for students with different learning styles, the following points should be considered:

- For active learners, a more detailed animation involving details of each step and corresponding code should be available.
- For reflective learners, along with the animation, detailed and appropriate captions explaining the process should be provided.
- Sequential learners will benefit more from the code, therefore detailed code and highlighting of the current path of execution should be more focused when designing the algorithm animation for sequential learners.

We grouped the participants into different clusters using VBMGS and analyzed the correlation of their behavior on paper-based test and computer based test. Our result showed that participants in cluster 3 (characterized by a preference for captions over code) have better performance on the inference test and participants in cluster 1 (characterized by a preference for code over captions) had better visualization capability. Participants in both clusters have similar performance on working memory. Participants in cluster 3 showed high reading span. These participants also had high correlation with the fixation duration at captions. Thus from our results we can say that while designing a visualization for participants in cluster 3 detailed textual description should be emphasized.

The results to date of the paper-based and computer-based tests are not yet clear enough to draw guidelines for designing algorithms. Nevertheless, we believe that these tests give insight into the perceptual, cognitive and attentional skills of participants. We recommend further studies with a large number of participants to derive conclusions on these tests.

This study was conducted as a pilot study for research in this area. This study provided guidelines for designing an effective algorithm animation by considering the impact of individual differences. For future work we recommend conducting this study on a larger scale in order to obtain increased statistical power. This will also help us to find the reasons for correlations on paper-based test and computer-based test. We also recommend evaluating the effect of popup questions by having a mix of low level questions and high-level questions and then looking at the user's gaze patterns.

References

[Agarwal07] M. Agarwal. Viewing Behavior Model Graphs (VBMG) For Characterizing Users Viewing Behavior In Program Visualizations. M.S Thesis, The University of Georgia, 2007.

[ASL01] Eyenal (Eye-Analysis) software manual Windows version for user with ASL Series 5000 and ETS-PC Eye Tracking Systems, Applied Science Group, Inc 2001.

[Baecker81] R. Baecker and D. Sherman. Sorting Out Sorting. 30 minute color sound film. Dynamic Graphic Project, University of Toronto, 1981.

[Bednarik05] R. Bednarik, N. Myller, E. Sutinen and M. Tukiainen. Effects of experience on gaze behaviour during program animation. In P*roceedings of the 17th Annual Psychology of Programming Interest Group Workshop (PPIG'05),* pp.49-61, 2005.

[Bednarik06] R. Bednarik, N. Myller, E. Sutinen, and M. Tukiainen. Program Visualization: Comparing Eye-Tracking Patterns with Comprehension Summaries and Performance. In *Proceedings of the 17th Annual Psychology of Programming Interest Group Workshop* (PPIG'06), pp.68-82, 2006.

[Branch00] J. L. Branch. Investigating the information-seeking processes of adolescence. The value of using think alouds and think afters. *Library and Information Science Research* Vol.22, No.3, pp.371-392, 2000.

[Brown97] M. H. Brown & M. A Najork. Collabrative active textbooks. *Journals of Visual Languages Computing* Vol. 8, 453-486, 1997.

[Brown98] M. H. Brown, M. and M. A. Najork. “Algorithm Animation Using Ineractive 3D Graphics.” In *Software Visualization: Programming as a Multimedia Experience*, 119-135. Cambridge, MA: The MIT Prees, 1998.

[Burkhardt02] J. Burkhadt., F. Detienne., and S. Wiedenbeck. Object-oriented program comprehension: Effect of expertise, task and phase. *Empirical Software Engineering.* Vol.7, No.2, pp.115-156, 2002.

[Crosby90] M. Crosby, and J. Stelovsky. How do we read algorithms? A case study. *IEEE Computer*, Vol. 23, No.1, pp. 24-35, 1990.

[Curry83] L. Curry. An Organization of Learning Styles Theory and Constructs. ERIC Document pp.235 185, 1983.

[Denning89] P. J. Denning. A debate on teaching computing science, *Communications of the ACM*, Vol.32, No.12, pp.1397-1414, Dec. 1989.

[Domingue98] J. Domingue and P. Mulholland. An effective web-based software visualization leaving environment. *Journals of Visual Languages Computing* Vol . 9, 485-508, 1998.

[Duchowski03] A.T. Duchowski. *Eye Tracking Methodology – Theory and Practice*, Springer-Verlag, 2003.

[Ekstrom 76] R. Ekstrom, J. French, H. Harman, and D. Derman. *Manual for kit of factor- referenced cognitive tests*. Princeton, NJ: Educational Testing Service, 1976.

[Faraday96] P. Faraday and A. Sutcliffe. An Empirical study of Attending and Comprehending Multimedia Presentations. In *Proceedings of ACM Multimedia, Boston*, MA, USA, 1996.

[Felder 88] R. M. Felder, L. K. Silverman. Learning and Teaching Styles in Engineering Education, *Journal of Engineering Education* Vol. 78, No. 7, pp. 647-681, 1988.

[Fitts50] P. M. Fitts, R. E. Jones and J. L. Milton. Eye movements of aircraft pilots during instrument-landing approaches. *Aeronautical Engineering Review*, Vol 9, No.2, pp.24-29, 1950.

[Hansen02] S. Hansen, N. H. Narayanan & M. Hegarty. Designing educationally effective algorithm visualizations, *Journal of Visual Languages And Computing* Vol 3, No. 3, pp. 291-317, 2002.

[Helttula90] E. Helttula, A. Hyrskykari & K. J. Raiha. Principles of Aladdin and other algorithm animation systems. In Visual Languages and application (T.Ichikawa, E. Jungert & R. Korfhage, eds). Plenum Press, New York, pp 175-187, 1990.

[Holmquist00] S. Holmquist and N. H. Narayanan. Tightly Coupling Authoring and Evaluation in an Integrated Tool to Support Iterative Design of Interactive Hypermedia educational manuals. In *Proceedings of conference on Designing Interactive Systems*, New York City, New York, United States, ACM Press, 2000.

[Hundhausen00] C. D. Hundhausen and S. Douglas. Using Visualization to Learn Algorithms: Should Students Construct Their Own, or View and Expert's? *IEEE Symposium on Visual Languages*, Los Alamitos, California, pp. 21-28, 2000.

[Hundhausen02] C. D. Hundhausen and S. A. Douglas and J. T. Stasko. A meta-study of algorithm visualization effectiveness. *Journal of Visual Languages and Computing*, Vol.13, pp. 259-290, 2002.

[Kann97] C. Kann, R. W. Lindeman, & R. Heller. Integrating algorithm animation into learning environment. *Computers & Education* Vol. 28, pp. 223-228, 1997.

[Karsh83] R. Karsh & F. W. Breitenbacj. Looking at looking: the amorphous fixation measure. In *R. Groner, C Menz, D. Fisher & R.A. Monty, Eye Movements and Psychological Functions: International Views*. pp.53-64. Lawerence Erlbaum Associates, Hillsdale, NJ. 1983.

[Kehoe01] C. Kehoe, J. Stasko, and A. Taylor. Rethinking the evaluation of algorithm animations as learning aids: An observational study. *International Journal of Human-Computer Studies*, Vol.54, No.2, pp.265-284, February 2001.

[Ko03] A. J. Ko and B. Uttl . Individual differences in program comprehension strategies in unfamiliar programming systems. In IWPC'03:

P*roceedings of the 11th IEEE International Workshop on Program Comprehension, IEEE Computer Society*. Washington, DC, USA,175, 2003.

[Kolb85] D. A. Kolb. LSI Learning-Style Inventory: Self-Scoring Inventory and Interpretation Booklet, Boston: McBer and Company, 1985.

[Kolers81] Kolers, P.A., Duchnicky, R.L. & Ferguson,D.C. Eye movements measurement of redability of CRT displays. *Human Factors*. 23:517-527, 1981.

[Kraemer] E. Kraemer, M. Tudoreanu, A. Taylor. Why Jhonny Won't Visualize.

[Lahtinen98] S. P. Lahtinen, E. Sutinen & J. Tarhio. Automated animation of algorithms with Eliot. *Journal of Visual Languages and Computing Vol.* 9, pp. 337-349, 1998.

[Lawrence94] A. W. Lawrence, A. M. Badre & J. T. Stasko. Empirically Evaluating the use of Animations to Teach Algorithms. *Proceedings of the 1994 IEEE Symposium on Visual Languages*, pp. 48-54, October 1994.

[Letovsky86] Cognitive processes in program comprehension. In Papers presented at the first workshop on empirical studies of programmers on Empirical studies of programmers, Ablex Publishing Corp., Norwood, NJ, USA, 58-79.

[Littman86] D. C. Littman, J. Pinto, S. Letovsky & E. Soloway. Mental models and software maintenance. In Papers presentated at the first workshop on empirical studies of programmers on Empirical studies of programmers, Ablex Publishing Corp., Norwood, NJ, USA, pp. 80-98, 1986.

[McWhirter96] AlgorithmExplorer: a student centered algorithm animation system. In P*roceeding of the IEEE Symposium on Visual Languages*. IEEE Computer Society Press, Piscataway, NJ, pp. 174-181.

[Moreno04] A. Moreno, N. Myller, E. Sutinen & M. Ben-Ari. Visualizing Programs with Jeliot 3. *In Proceedings of the Working Conference on Advance Visual Interfaces (AVI 2004)*. ACM, pp. 373-376, 2004.

[Nanja87] M. Nanja and C. R. Cook. An Analysis of the On-Line Debugging Process. In *Empirical Studies of Programmers: Second Workshop*, G. M. Olson, S. Shepard, and E. Soloway, Eds. Norwood, NJ: Ablex, pp. 172-184, 1987.

[Narayanan02] N. H. Narayanan & M. Hegarty. Multimedia design for communication of dynamic information. *International Journal of Human ComputerStudies*, Vol. 57, No.4, pp. 279-315, 2002.

[Naps03] T. G. Naps, A. J Robling, S.Cooper, et al. Evaluating the educational impact of visualization. Working group reports from ITiCSE on Innovative and technology in computer science education, Thessaloniki, Greece, ACM Press, 2003.

[Nielsen02] J. Nielsen, T. Clemmensen, and C. Yssing. Geeting acessess to what goes on in peoples head? Reflections on the think-aloud technique. In *Proceedings of the Second Nordic Conference on Human Computer Interaction (NordiCH'02)*, ACM Press, New York, NY, USA, pp. 101-110, 2002.

[Price90] B. Price. A framework for the automatic animation of concurrent programs. Unpublished master's thesis, Department of Computer Science, University of Toronto, Canada, 1990.

[Price98] B. Price, R. Baecker & I. Small. An Introduction to Software Visualization. In S*oftware Visualizaation: Programming as a Multimedia Experience*, 3-27. Cambridge, MA: The MIT Press, 1998.

[Rayner98] K. Rayner. Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin. Vol.* 124, pp. 372-422, 1998.

[Reed06] B. N. Reed. B.N. Investigating Characteristics of Effective Program Visualizations: A Testing Environment And The Effect of Comparison Cueing And Exchange Techniques On Viewer Comprehension In Algorithm Animations, *M.S. Thesis. The University of Georgia*, 2006.

[Rhodes06] P. Rhodes, E. Kraemer and B. N. Reed, The Importance of Interactive Questioning Techniques in the Comprehension of Software

Visualizations, *ACM Symposium on Software Visualization (SoftVis'06)*, September 4-5,2006, Brighton,UK.

[Salvucci00] D. D. Salvucci & J. H. Goldberg. Identifying fixations and saccads in eye-tracking protocols. In *Proceedings of the Eye tracking Research and Application Symposium*. Pp. 71-78. NY: ACM Press, 2000.

[Saraiya04] P. S. Saraiya, C. A. Shaffer, D. S. Mc Crickard & C. North C. Effective Features of Algorithm Visualizations. In *Proceedings of the 35th SIGCSE technical symposium on Computer science education*. Norfolk, Virginia, USA pp: 382 – 386. 2004.

[Scaife96] M. Scaife & Y. Rogers. External cognition: how do graphical representations work? I*nternational Journal of Human-Computer Studies*, Vol. 45, pp. 185-213, 1996.

[Sibert00] L. E. Sibert, R. J. Jacob. Evaluation of eye gaze interaction. In: *CHI* 2000, ACM Press, pp. 281-288, 2000.

[Soloway88] E. Soloway, R. Lampert, S. Letovsky, D. Littman & J. Pinto 1988. Desigining documentation to compensate for delocalized plans. *Communication of ACM*, Vol. 31, No. 11, pp no. 1259-1267, 1988.

[Stasko93] J. Stasko, A. Badre, & Lewis. Do algorithm animations assist learning? An empirical study and analysis. Proceedings of ACM INTERCHI'93 Conference on Human Factors in Computing Systems (pp. 61-66). New York: ACM, 1993.

[Stasko95] J. Stasko . POLKA animation designer's package, Georgia Institute of Technology, Atlanta, GA, 1995.

[Stasko98] J. Stasko. Smooth, Continuous Animation for Portraying Algorithms and Processes. In Stasko, J., Domingue, J., Brown, M. H., & Price, B. A.(1998). *Software visualization: programming as a multimedia experience*, Ch.9, pp103-118. Cambridge, MA: MIT Press.

[Stroop35] Stroop, J. R. Studies of interference in serial verbal reactions. *Journal of Experimental Psychology*, Vol. 18, pp. 643-662, 1935.

[Tudoreanu02] M. Tudoreanu, R. Wu, A. Hamilton-Taylor, E. Kraemer. Empirical Evidence that Algorithm Animation Promotes Understanding of Distributed Computations, *Proceeding, IEEE 2003 Symposia on Human Centric Computing Languages and Environments*, Arlington, VA, pp.236-243, Sep. 3-6, 2002.

[von Mayrhauser96] A. von Mayrhauser, A. M. Vans. On the Role of Hypotheses during Opportunistic Understanding While Porting Large Scale Code**.** In *Proceedings of the IEEE Fourth Workshop on Program Comprehension* 1996.

Appendix A

Questions for Pre-Test/Post-Test Quicksort Algorithm

Pre-Test Questions:

1. Which best describes how quicksort works?
    a) It partitions an array into two subarrays and uses the median value of the array as the pivot.
    b) For each elementx, in the array, move x to the index equal to the number of elements that are less than x.
    c) It partitions the array into two subarrays and sorts the subarrays independently.
    d) It swaps adjacent items that are out of order.

    Answer: c
    Classification: Comprehension

2. How is the pivot used?
    a) To identify the largest element of the array.
    b) To identify the smallest element of the array.
    c) To identify the median element of the array.
    d) To separate the elements of the array into two subarrays.

    Answer: d
    Classification: Comprehension

3. Given the sequence 7 8 6 2 1 9 4 3. If 3 is chosen as the pivot, which of the following could be the new order after the first call to the partition function?
    a) 2 1 3 7 8 9 4 6
    b) 7 6 2 1 8 4 3 9
    c) 7 8 6 2 3 1 9 4
    d) 1 7 8 6 2 3 9 4

    Answer: a
    Classification: Comprehension
4. When does the worst-case time for quicksort occur for an array of n elements?
    a) When the pivot is always the largest or smallest element in the active partition.
    b) When the input size is a power of 2.
    c) When the partition splits the array into 2 subarrays of equal lengths.
    d) There is no predictor for worst-case time.

    Answer: a
    Classification: Comprehension

5. When does the best-case time for quicksort occur for an array of n elements?
    a) When the pivot is always the largest or smallest element in the active partition.
    b) When the input size is a power of 2.
    c) When the partition splits the array into 2 subarrays of equal lengths.
    d) There is no predictor for best-case time.

Answer: c
Classification: Comprehension

6. The quicksort algorithm can best be described as:
   a) selective
   b) recursive
   c) iterative
   d) abstract

   Answer: b
   Classification: Comprehension

7. During a run of the partition function each number is compared to:
   a) its neighbor
   b) all other numbers
   c) itself
   d) the pivot

   Answer: d
   Classification: Comprehension

8. The outcome of partitioning is:
   a) to place all numbers in sorted order
   b) that no number in the lower partition is larger than any number in the higher partition
   c) to place half of the numbers into the left partition
   d) to place all numbers larger than pivot in sorted order

   Answer: b
   Classification: Comprehension

Post-test Questions:

Please answer the following questions based on the version of the quicksort algorithm depicted in the animation.

9. Which best describes the correct order of events of the version of quicksort you just viewed.

   I. Call quicksort on the higher partition.
   II. Compare elements to the pivot. If it is less than or equal in value, then swap element into the lower section of the partition.
   III. Select a pivot.
   IV. Call quicksort on the lower partition.
   V. Swap the pivot into the position between the lower and higher partitions.

   a) I, II, III, IV, V
   b) III, II, V, IV, I
   c) III, II, I, IV, V
   d) II, III, IV, I, V

Answer: b
Classification: Comprehension

10. Which element is chosen as the pivot in the active partition?
    a) The leftmost element
    b) A Random element
    c) The middle element
    d) The rightmost element

    Answer: d
    Classification: Comprehension

11. When is the pivot swapped?
    a) When a value less than or equal to the pivot value is found.
    b) When a value greater than the pivot value is found.
    c) At the end of partitioning a subset of the array.
    d) None of the above

    Answer: c
    Classification: Comprehension

12. The pivot is swapped with ____________________.
    a) the first element in the lower partition.
    b) the last element in the lower partition.
    c) the first element in the higher partition.
    d) the last element in the higher partition.

    Answer: c
    Classification: Comprehension

13. The comparison of an element with the pivot is done in which method(s)?
    a) quicksort()
    b) partition()
    c) swap()
    d) a and b

    Answer: b
    Classification: Comprehension

14. What two objects are being compared in the partitioning step?
    a) The element at 'firstHigh' and the element at 'pivot'
    b) The element at 'begin' and the element at 'pivot'
    c) The element at 'firstHigh' and the element at 'findLow'
    d) The element at 'findLow' and the element at 'pivot'

    Answer: d
    Classification: Comprehension

15. Swaps can occur between ___________________.
    a) the element at 'firstHigh' and the element at 'findLow'
    b) the element at 'begin'+1 and the element at 'firstHigh'
    c) the element at 'firstHigh'+1 and the element at 'pivot'

d) the element at 'findLow' and the element at 'firstHigh'+1

Answer: a

Classification: Comprehension

16. Assume that the array to be sorted initially contained the following values: 5 7 8 2 9 6. Which of the following will be the higher partition after one invocation of quicksort?
    a) 7 9 8
    b) 6 7 9 8
    c) 7 8 9
    d) 6 7 8 9

    Answer: a

    Classification: Comprehension

17. Given the array 8 3 7 5 1 6 2 4. Which of the following represents the contents of the new array after one invocation of quicksort?
    a) 1 2 3 4 6 8 5 7
    b) 4 3 2 1 5 6 7 8
    c) 4 3 7 5 1 6 2 8
    d) 3 1 2 4 8 6 7 5

    Answer: d

    Classification: Comprehension

18. Which best describes how quicksort works?
    a) It partitions an array into two subarrays and uses the median value of the array as the pivot.
    b) For each element,x, in the array, move x to the index equal to the number of elements that are less than x.
    c) It partitions the array into two subarrays and sorts the subarrays independently.
    d) It swaps adjacent items that are out of order.

    Answer: c

    Classification: Comprehension

19. How is the pivot used?
    a) To identify the largest element of the array.
    b) To identify the smallest element of the array.
    c) To identify the median element of the array.
    d) To separate the elements of the array into two subarrays.

    Answer: d

    Classification: Comprehension

20. When does the worst-case time for quicksort occur for an array of n elements?
    a) When the pivot is always the largest or smallest element in the active partition.
    b) When the input size is a power of 2.
    c) When the partition splits the array into 2 subarrays of equal lengths.
    d) There is no predictor for worst-case time.

    Answer: a

    Classification: Comprehension

21. When does the best-case time for quicksort occur for an array of n elements?
    a) When the pivot is always the largest or smallest element in the active partition.
    b) When the input size is a power of 2.
    c) When the partition splits the array into 2 subarrays of equal lengths.
    d) There is no predictor for best-case time.

    Answer: c
    Classification: Comprehension
22. The quicksort algorithm can best be described as:
    a) selective
    b) recursive
    c) iterative
    d) abstract

Answer: b
Classification: Comprehension

23. During a run of the partition function each number is compared to:
    a) its neighbor
    b) the pivot
    c) all other numbers
    d) itself

    Answer: b
    Classification: Comprehension

24. The outcome of partitioning is:
    a) to place all numbers in sorted order
    b) that no number in the lower partition is larger than any number in the higher partition
    c) to place half of the numbers into the left partition
    d) to place all numbers larger than pivot in sorted order

    Answer: b
    Classification: Analysis

Appendix B

Pop up Questions

1. Why did 70 and 20 just swap?
   a. because the partition() method ended
   b. because findLow > findHigh
   c. because findLow <= firstHigh
   d. because findLow <= Pivot

Answer: d

2. All of the numbers to the left of 40
   a. are in the lower partition
   b. are in the higher partition
   c. are greater than the pivot
   d. are sorted

Answer: a

3. The pivot(20) was just swapped, when will it move again
   a. at the end of quicksort()
   b. at the end of partition()
   c. never
   d. it cannot be determined

Answer: c

4. Why was the quicksort call skipped
   a. end <=begin
   b. end > begin
   c. the pivot is out of range
   d. it was not skipped

Answer = a

5. Which number will be the pivot next?
   a. 30
   b. 40
   c. 80

d. 60

Answer = d

6.Why did 60 and 80 just swap?

a. because the partition() method ended
b. because findLow > findHigh
c. because findLow <= findHigh
d. because findLow <= pivot

7. Which subarray of numbers will be sorted next?

a. the higher partition
b. the lower partition
c. the entire array
d. the faded portion

Answer = b

8. Which variable will be the next to swap with the pivot?

a. begin
b. firstHigh
c. firstLow
d. both a) and c)

Answer = c

APPENDIX C

Graphs of visual pattern

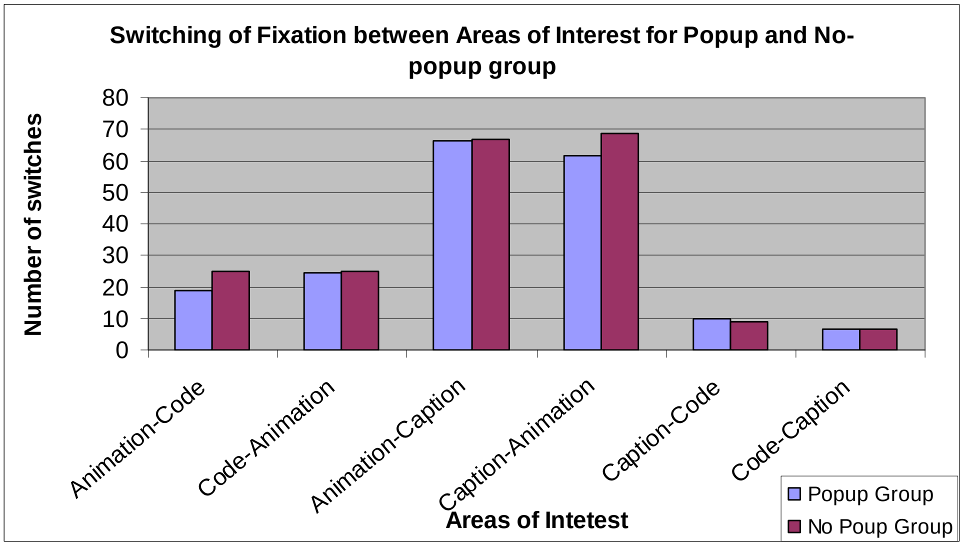

Figure C.1: Switching of fixation from one area of interest to another for Popup and No-popup group

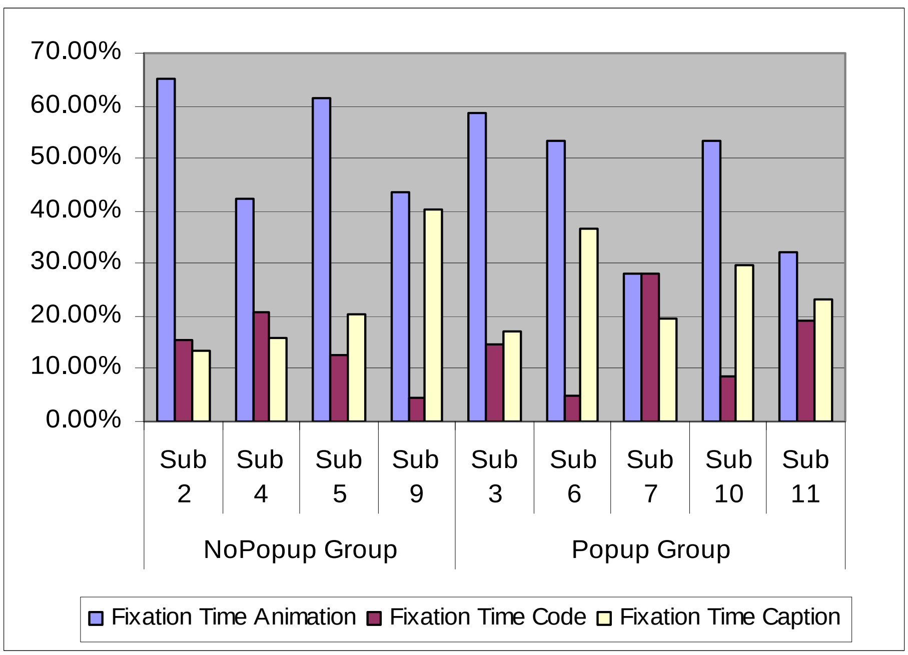

Figure C.2: Viewing pattern of all participants

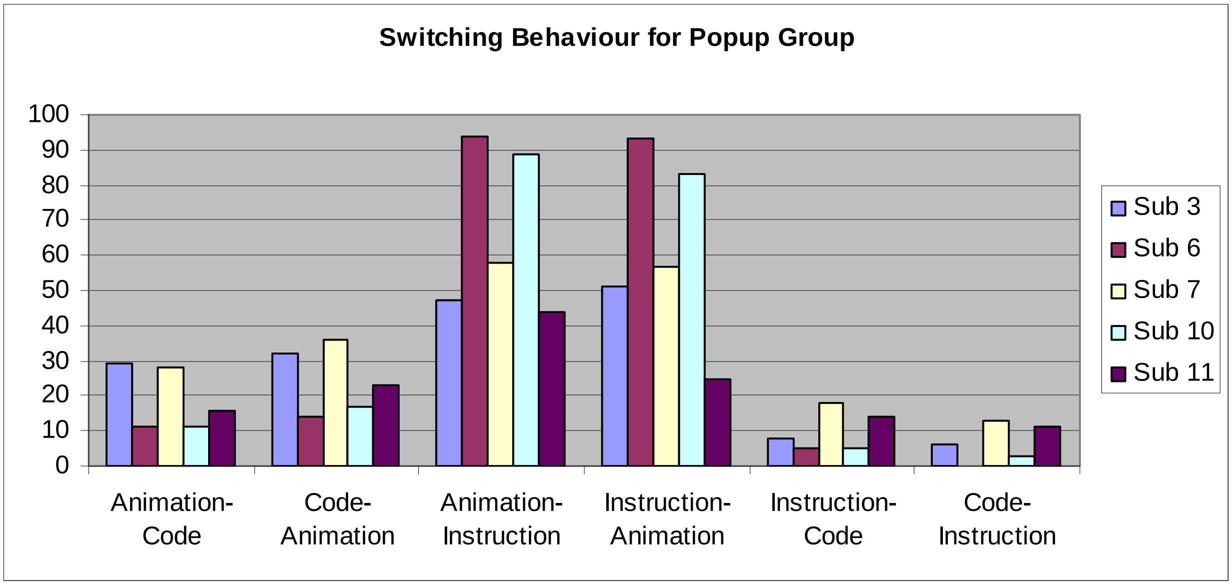

Figure C.3: Switching Behavior of each subject in Popup group

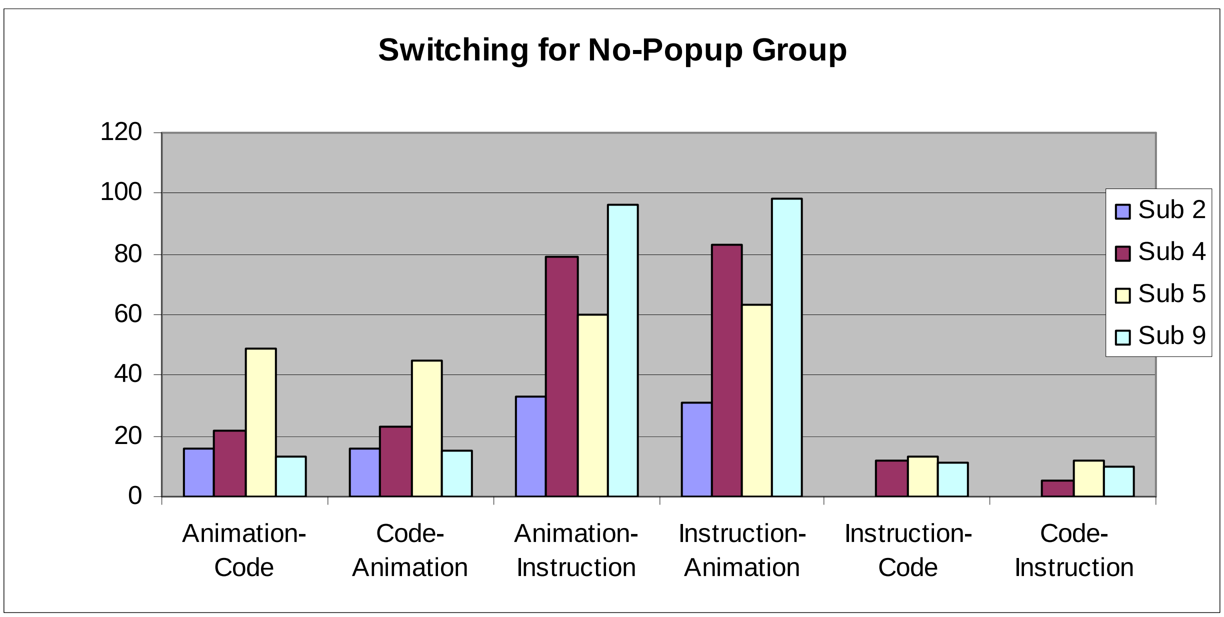

Figure C.4: Switching Behavior of each subject in No-Popup group

Appendix D

Scores and correlation for assessments

Table D.1: Scores of participants on paper based assessments

		Paper Based Test			
Group	**Subject #**	**Inference Test – RL 3**	**Fig Class I-3**	**Surface Dev Vz-3**	**Size Span Test**
Popup	3	11.5	105	18	3
Popup	6	13.5	127	1.5	4
Popup	7	11	118.5	22.5	5.5
Popup	10	17	115.5	28.5	4
Popup	11	10	114	25.5	3.5
NoPopup	2	13	104	30	4
NoPopup	4	15.5	119	30	4
NoPopup	5	9.5	158.5	18.5	5.5
NoPopup	9	17	130.5	24	4

Table D.2: Correlation of Paper based assessment score with fixation duration

Overall	**RL-3**	**Fig Class I-3**	**Surface Deve Vz-3**	**Size Span Test**
Fixation Time for Animation	0.058	0.098	-0.162	-0.159
Fixation Time for Code	-0.490	-0.290	0.410	0.297
Fixation Time for Instruction	0.498	0.309	-0.452	-0.080

Table D.3: Scores of participants on computer based assessments

		Computer Based Assessment			
Group	**Subject #**	**R-Span**	**S- Span**	**O-Span**	**Color Stroop**
Popup	3	37	26	47	80
Popup	6	51	23	23	78
Popup	7	65	22	52	40
Popup	10	45	37	62	80
Popup	11	0	25	3	80
No-Popup	2	46	23	44	80
No-Popup	4	40	42	48	80
No-Popup	5	3	19	58	76
No-Popup	9	64	31	62	78

Table D.4: Correlation of Computer based assessments with fixation duration

Overall	R-Span	S-span	O-span	Color Stroop
Fixation Time for Animation	-0.12209	-0.15866	0.292122	0.571119799
Fixation Time for Code	-0.09413	-0.06897	-0.13505	-0.629590851
Fixation Time for Instruction	0.321825	0.091586	0.013428	0.137714445