An Experiment on Short-Term Effects of Animated versus Static Visualization of Operations on Program Perception

Seppo Nevalainen  
Department of Computer Science  
University of Joensuu  
P.O.Box 111  
Joensuu, Finland  
snevalai@cs.joensuu.fi

Jorma Sajaniemi  
Department of Computer Science  
University of Joensuu  
P.O.Box 111  
Joensuu, Finland  
saja@cs.joensuu.fi

ABSTRACT
Empirical evaluation of visualizations has so far been typically carried out by measuring the performance of participants that have been shown the visualization in relation to control group by grading programming tasks. Such studies tell little about what effects take place during the visualizations and how these effects build up into the learning effects. To address these issues, we are carrying out a series of experiments using visualization tools whose long-term effects are known.

This paper presents a model of the cognitive phenomena that take place during visualization viewing sessions and describes an experiment where two visualization tools that differed in the amount of animation were used. The results show that even when participants were provided with rich visual information in the form of animations, they resorted heavily to the textual cues present on the screen. As the animation proceeded, participants started to follow visual cues more closely. However, if the names of the variables involved in the current operation were not in the visually cued area, participants tended to search for them in the program code. In the absence of animation, participants were more eager to browse the code in general.

1. INTRODUCTION
One application area for visualizations in computer science is teaching of computer programming, where the goal is to help students in evolving from programming novices to experts. Achieving this goal is challenging, because programs deal with abstract entities—formal looping constructs, pointers going through arrays etc.—that have little to do with everyday issues, and that make learning to program difficult for many students. Visualization methods that help students to better understand and conceptualise these abstract entities and their behavior are needed. This need has been noticed, and numerous visualization and animation tools have been developed (e.g., ALADDIN [10], ANIMAL [19], Eliot [14], DISCOVER [17], Jeliot [8], ZEUS [1]). Visualizations are not, however, a panacea; see for example [12]. Before a visualization method or tool can be utilized in teaching, it needs to be properly evaluated.

So far, empirical evaluation of visualizations has been mostly concentrating on the long-term effects of the visualizations. Evaluation has typically been carried out by measuring the performance of participants that have been shown the visualization in relation to other participants, who have been shown similar material without visualising it. Performance has been measured either by comparing the scores of participants after the viewing, or by comparing the change in the scores of the participants between before and after the viewing. These post-tests and pre- versus post-tests of participants’s performance have resulted in a body of evidence suggesting that properly designed visualizations can have beneficial effects on the learning of the students [4, 9, 12, 13]. These evaluations do not, however, provide clear insight into the possible short-term effects of visualizations and the relation between short-term effects and long-term effects. In other words, the results tell whether an individual visualization tool is helpful or unhelpful in achieving better learning outcomes, but they do not tell much about what effects take place during the use of the visualization tool and how these effects build up into the long-term learning effects.

To address these issues, we are carrying out a series of experiments using visualization tools whose long-term effects are known. We have developed a model of the cognitive phenomena that take place during visualization viewing sessions and our research goals are (i) to find out in detail how various aspects of visualizations affect these phenomena and what interactions these phenomena have, and (ii) to un-
Learning to program involves the development of various mental structures, e.g., programming language syntax and semantics, general programming principles like how loops are used to initialize arrays, specific constructs used in specific programs etc. In the following, we will make a clear distinction between program knowledge that concerns a single program, e.g., what is the purpose of the variable count in a program, and general programming knowledge that is needed in authoring new programs, e.g., for what purposes counters are typically used in programs. In programming education, programming knowledge is usually introduced through program knowledge: students are given example programs and they are supposed to construct the general programming principles by themselves. When studying a new example, the student develops program knowledge directly based on the program code but the development of general programming knowledge requires deeper mental elaboration of the new information and may take longer time.

When a visualization tool is used in teaching programming, the content of the visualization plays a large role in the question whether the tool benefits learning or not. Other, possibly even more significant factors determining the beneficiality of the visualizations are the level of engagement of the student with the visualization tool, and the way the visualizations succeed in attracting the interest of the student [12]. The use of an animation tool results in different knowledge contents, i.e., different mental models, when compared to students not using the tool. On short-term these changes may be related both to general programming knowledge and to specific knowledge of the studied program. On long-term, details of individual programs will be forgotten, and differences concern mainly the more general programming knowledge. The final long-term knowledge is not only affected by the short-term programming knowledge but also by short-term knowledge of the studied programs [6, 7, 16, 18].

Finally, if animation is to have any effect at all, students have to follow it, i.e., the focus of visual attention has an effect on the resulting mental models. Even though the location of visual attention is not always equal to the point of person's interest, it provides strong indication on what a person chooses to target her interest on [5] and, moreover, a visualization that does not get visual attention is unlikely to affect the resulting mental model.

Figure 1 depicts the above phenomena and their interactions. The short-term mental models may be affected by the content and visual appearance of the visualizations used in the animation but only if they get visual attention. On the other hand, the focus of visual attention may depend on the part of the visualization that is needed for the creation of a correct and meaningful short-term mental model.

Similarly, the effects of engagement (viewing a ready-made animation, entering input etc) are two-fold: engagement affects the level of commitment to the task which has direct
effects to the strength of resulting mental models, but it also
directs visual attention to specific points of the visualization
and thus strengthens their effects on the mental models.

In algorithm animation, the main objective is to teach
students how an individual algorithm works. Accordingly,
the more important part of the resulting short-term mental
model is program knowledge. On the other hand, in pro-
gram animation the opposite is true: even though a single
program is animated at a time, the main objective is to in-
crease generally applicable programming knowledge both at
short term and at long term.

Research on the effects of animation has so far looked
mostly at the long-term effects. Moreover, the role of visual
attention and its relationship with visual characteristics has
not been studied at all. To address these issues in detail,
we are carrying out a series of experiments (i) to find out in
detail how various aspects of visualizations affect these
phenomena and what interactions these phenomena have,
and (ii) to untangle how these phenomena are related to
short-term and long-term effects of visualizations. We do
this with the help of empirical experiments where the effects
of visualization techniques are studied in detail.

2.2 The Research

We are conducting our research using the PlanAni pro-
gram animator [21] whose long-term effects, when compared
with Turbo Pascal debugger, are known [3, 22]. The pur-
pose of PlanAni is to consolidate knowledge about roles of
variables [20], which forms the programming knowledge of
Figure 1 in our case.

Roles of variables describe stereotypical dynamic behav-
iors of variables, e.g., a variable whose value does not change
after its initialization is said to have the role of a fixed value.
A set of eleven roles covers 99% of all variables in novice-
level programming which makes them practical to be used in
elementary programming education. Roles give names to
various classes of variables and stress a data-based view
of programming: variables are entities that have specific
responsibilities, and these responsibilities are implemented
with stereotypical code sequences. Table 1 lists five roles
that are used in the experiment described in this paper.

Figure 2 is a screen shot of the PlanAni user interface.
The program code is located on the left side of the window,
variable visualizations on the right side of the program code,
and a separate area for input and output at the bottom of
the window, above the control buttons. The currently ac-
tive action of the program code is highlighted with a color
background that flashes every time control moves to a new
location. PlanAni displays also notification of each program
action with a message box that appears on the center of
the screen. The messages include variable creation (e.g., “cre-
ating a gatherer called sum”), operations (“comparing count
with zero”), and control constructs (“entering a loop”).

In PlanAni, variables are visualized with role images and
the animation of operations on variables depends on their
role. The role images are based on metaphors that repre-
sent the salient, stereotypical features of variables’ behavior,
e.g., a dog for the role of follower, a box for the role of gath-
erer etc. Thus PlanAni stresses the role concept both in
visualization of variables and in animation of operations.

The beneficial long-term effects of visualizations in
PlanAni on learning found in [3, 22] are based on measuring
the performance of students after they had used the visual-
ization tool during four weeks, for half an hour each week.
PlanAni users were found to have better programming skills
and their mental representation of (not animated) programs
were different form that of Turbo Pascal debugger users and,
in fact, similar to that of good code comprehenders.

In our research, we focus on what happens during individ-
ual occasions of learning when a student uses a visualization
tool to understand a computer program. We are interested
in the development of students’ mental models concerning
the studied programs and the general programming knowl-
dge concerning the roles. In a series of experiments, we
use the content of visualizations and the level of viewer’s
engagement as the varied factor, and gather information of
the effects of this variation on the visual attention and the
mental models. So far, we have studied the effect of the con-
tent of visualization on programming knowledge [23] and the
effect of the content of visualizations on visual attention and
mental model of the studied program [15].

The experiment on the effect of the content of visualiza-
tions on short-term mental model of roles [23] was carried
out by comparing empirically the effects of PlanAni’s origi-
nal role images versus neutral control images. In this exper-
iment, engagement was not varied and visual attention was
not measured. The main interest was whether the content
of visualization—in this case, the individual role images—
makes a difference in the learning outcome and, indeed, this
was the fact: the results indicated that the use of the origi-
nal role images enhanced learning of role knowledge when
compared with neutral control images. In this experiment,
the target of visual attention was not measured but the tasks
suggested similar visual behavior for both groups. Thus it
seems evident that the difference is more related to the con-
tent of the visualizations than to the effect the content had
on directing visual attention.

The effect of the content of visualizations on visual attention,
and short-term mental model of a studied program [15]
was studied in an experiment where visualizations of vari-
bles were either graphical and animated (PlanAni) or tex-
tual and static (Turbo Pascal programming environment
version 5.5). The graphical visualizations depicted the roles
of the variables. The results showed clear difference in the
targeting of visual attention between the textual visualiza-

<table>
<thead>
<tr>
<th>Role</th>
<th>Informal description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed value</td>
<td>A variable initialized without any calculation and not changed thereafter.</td>
</tr>
<tr>
<td>Stepper</td>
<td>A variable stepping through a systematic, predictable succession of values.</td>
</tr>
<tr>
<td>Most-recent holder</td>
<td>A variable holding the latest value encountered in going through a succession of values, or simply the latest value obtained as input.</td>
</tr>
<tr>
<td>Most-wanted holder</td>
<td>A variable holding the best or otherwise most appropriate value encountered so far.</td>
</tr>
<tr>
<td>Gatherer</td>
<td>A variable accumulating the effect of individual values.</td>
</tr>
</tbody>
</table>
tion tool and the graphical visualization tool. With the graphical tool, the participants targeted their visual attention to variables much more than with the textual tool. With the graphical tool, the increase of visual attention to variables increased the proportion of high-level information in program summaries and decreased the proportion of low-level code-related information.

In the current experiment, the content of visualizations is graphical and depicts the roles of variables. The visualizations of operations either contain animations or are static and we investigate the interaction between presence/absence of animation, visual attention, and short-term mental model of both variable roles (programming knowledge) and the studied program (program knowledge).

3. EXPERIMENT

In order to study the differences in short-term effects between animated and static visualizations of operations on variables, we conducted an experiment where two different versions of PlanAni program animator were used. The experiment concentrates on the analysis of possible differences in the locations of participants’ visual attention and in participants’ mental models concerning variable role knowledge and studied programs between the two versions of the animator.

3.1 Method

The experiment was a between-subject design with one independent variable (version of the visualization tool) and several dependent variables (locations of participant’s gaze, participant’s post-test score of variable role knowledge, and the program summary provided by the participant). Locations of gaze were recorded using an eye-tracking camera [24]. Knowledge on variable roles was measured using a post-test from [23]. The program summaries were analyzed using Good’s program summary analysis scheme [7] with the additional categories presented in [2]. The participants were divided into two groups, animation group and static group, on the basis of a pre-test score (used earlier in [23]) so that the average scores of the groups were as equal as possible.

3.1.1 Participants

Sixteen participants, 11 male and five female, took part in the experiment. The participants were students who had taken a first-year programming course in last eighteen months and continued their studies thereafter. The participants received a fee of 15 euros.

3.1.2 Materials

In the first phase, participants were shown a short 18 minutes video of roles of variables that introduced five roles (fixed value, stepper, most-recent holder, most-wanted holder, and gatherer) in more detail.

In the second phase, a pre-test on role knowledge was carried out. The pre-test consists of descriptions of all roles and examples of their use, and three small programs with 11 variables, whose roles participants were asked to determine. For the purpose of this experiment, the programming language in the pre- and post-tests was changed from Pascal to Java, and the programs were modified to limit the roles to the five roles shown in the video presentation of the first phase.

In the third phase, participants studied two simple Java programs. Program lengths were 63 and 58 lines, empty lines omitted. Programs consisted of a short main program, and a class that included constructor and three methods in first program, and constructor and two methods in second program. The programs were designed so that they were similar to the more difficult programs used in the programming course the participants had taken. Participants entered predefined inputs for the programs. The use of fixed
inputs enabled a participant to focus her attention to understanding the program, instead of wondering what inputs would be proper to the programs.

The programs were viewed with PlanAni program animator. For the purpose of the experiment, a modified version of PlanAni was programmed. In the modified version of PlanAni the animations for operations on variables and the notifications were removed, and only the role images for the variables and the values the variables hold during execution were displayed. The operations were executed automatically, and viewer could halt the execution during a request for user input only. Animation group used the original animated version of PlanAni, and static group used the modified static version. Both versions were prepared so that participants were able to execute each program once, step by step. This limitation was used because we wanted to minimise differences in the amount of time used for the viewing between the groups and between individual participants. The PlanAni window covered the entire screen of a 1280 * 1024 resolution display.

In the fourth phase, a post-test on role knowledge was carried out. The post-test consisted of three small programs with 13 variables, whose roles participants were asked to determine. In the post-test, the descriptions of all roles and examples of their use were replaced with a list of the names of all variable roles after each program.

In the fifth phase, participants were asked to evaluate PlanAni with an evaluation form including Likert scale questions and open questions about the tools and their use. In the Likert scale questions, participants were asked to use a scale of 1-5 (1 = totally disagree, 5 = totally agree) to statements concerning five characteristics of the visualisations: originality, pleasure, salience, understandability, and usefulness. For example, the understandability of the visualisations was evaluated by proposition “I found this representation easy to understand”. These characteristics were derived from experiments carried out by Hübischer-Younger and Narayanan [11] who used them to characterise student visualisations of algorithms. In the open questions, participants were asked to report what issues the two visualisations did and did not highlight. The evaluation form included also a possibility for free commentary.

3.1.3 Procedure

Participants were run individually. First, the participant was informed about the phases of the experiment, and the tasks of the participant during the phases. Then, the participant viewed an 18 minutes video of roles of variables.

After this, the participant was given 15 minutes to perform the roles of variables pre-test. At the point of ten minutes, the participant was informed that she had five minutes left. During this time the participant was free to browse the role descriptions and examples, and the three programs she was asked to assign the roles to.

Then, the participant was seated in front of a computer monitor that has an eye-tracking camera embedded in the panels. The functionality of the eye-tracking camera and the procedure of measuring the movement of her eyes was explained to the participant. After this, the calibration of the camera was carried out. During this time the post-test score of the participant was scored and she was assigned to appropriate group. The participant was advised about the locations of all available information on the screen for the PlanAni version she was going to use to view the two programs.

After this, the viewing of the two programs was started. The participant had an unlimited time to study each program. After the participant had finished studying a program, the program was dismissed from the screen, and she was instructed to give a written description of the program. Again, the time to do this was not limited, and the participant was not instructed in any way on what the program description should contain. The first program was used to familiarise the participant with the PlanAni, and data from the second program only was analysed.

After this, the participant was given ten minutes to perform the roles of variables post-test. At the point of eight minutes, the participant was informed that she had two minutes left. After the post-test, the participant was asked to evaluate the PlanAni version she used. Again, the time to do this was not limited. The length of an experiment session was approximately two hours.

3.2 Results

A pre-test on variable role knowledge was used to divide the participants into two groups. The maximum in the pre-test is 11 points. The mean score of participants in animation group was 10 points (Standard deviation 0.93), and in static group 9.88 points (Standard deviation 1.46).

The maximum points in the post-test on variable role knowledge is 13. The mean score of participants in animation group was 12 points (Standard deviation 1.07), and in static group 11.25 points (Standard deviation 2.66).

Independent samples t-test did not show significant difference on the scores between the groups in the post-test.

The mean time used to study a program by participants in animation group was 19 minutes and 9 seconds. Standard deviation was two minutes. In static group, mean time to study a program was 12 minutes and 27 seconds, and the standard deviation was 6.6 seconds.

In order to investigate the distribution of visual attention, the screen was divided into five areas. The code area, the variable area, and the IO-area were formed by taking the smallest bounding box that includes the symbols within the areas. The notifications area was formed by taking the bounding box surrounding largest notifications message box. Other parts of the screen formed the fourth area. The mean proportions of viewing times on these four areas are presented in Table 2.

A Two-way Mixed-Design ANOVA on absolute viewing times (excluding notifications) was carried out. Only the main effect of screen area was statistically significant ($F(3, 42) = 53.105, p < 0.001$). For animation group, a One-Way Within-Subjects ANOVA with Bonferroni adjustment showed significant differences between absolute viewing times spent on different areas of the screen except code area and notifications, variable area and IO area, and variable area and notifications ($p < 0.05$). For static group differences were significant except between variable area and IO area. To analyze the possible differences between the groups in the distribution of time among the parts of the screen (excluding notifications), One-Way Between-Subjects ANOVAs for proportional viewing times were carried out. The tests showed that the difference between animations group and static group is significant in the proportional viewing times on code area and IO area ($p < 0.05$).
Table 2: Proportional and absolute (minutes, seconds) mean viewing times on the three areas of the screen.

<table>
<thead>
<tr>
<th>Code</th>
<th>Screen Area</th>
<th>Animation</th>
<th>Static</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Proportional</td>
<td>SD</td>
<td>Absolute</td>
</tr>
<tr>
<td>COD</td>
<td>Code</td>
<td>44.05</td>
<td>8.19</td>
</tr>
<tr>
<td>VAR</td>
<td>Variables</td>
<td>14.96</td>
<td>3.92</td>
</tr>
<tr>
<td>NOT</td>
<td>Notifications</td>
<td>28.67</td>
<td>10.28</td>
</tr>
<tr>
<td>IO</td>
<td>Input and Output</td>
<td>9.77</td>
<td>4.04</td>
</tr>
<tr>
<td>OTH</td>
<td>Other</td>
<td>2.55</td>
<td>1.18</td>
</tr>
</tbody>
</table>

Table 3: Mean proportions of IT categories used in program summaries.

<table>
<thead>
<tr>
<th>Code</th>
<th>Information Type</th>
<th>Animation</th>
<th>Static</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>SD</td>
<td>Mean</td>
</tr>
<tr>
<td>FUN</td>
<td>4.45</td>
<td>4.96</td>
<td>6.90</td>
</tr>
<tr>
<td>ACT</td>
<td>15.64</td>
<td>6.22</td>
<td>22.39</td>
</tr>
<tr>
<td>OPE</td>
<td>4.91</td>
<td>6.03</td>
<td>6.78</td>
</tr>
<tr>
<td>SHI</td>
<td>6.41</td>
<td>6.92</td>
<td>1.85</td>
</tr>
<tr>
<td>SLO</td>
<td>5.44</td>
<td>6.90</td>
<td>5.33</td>
</tr>
<tr>
<td>DAT</td>
<td>49.58</td>
<td>15.80</td>
<td>46.21</td>
</tr>
<tr>
<td>CON</td>
<td>5.24</td>
<td>5.72</td>
<td>4.09</td>
</tr>
<tr>
<td>ELA</td>
<td>0.00</td>
<td>0.00</td>
<td>2.75</td>
</tr>
<tr>
<td>MET</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>IRR</td>
<td>5.60</td>
<td>8.58</td>
<td>3.78</td>
</tr>
<tr>
<td>UNC</td>
<td>0.00</td>
<td>0.00</td>
<td>0.96</td>
</tr>
<tr>
<td>INC</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>CUT</td>
<td>2.70</td>
<td>3.77</td>
<td>0.00</td>
</tr>
<tr>
<td>HIG</td>
<td>76.09</td>
<td>12.69</td>
<td>77.31</td>
</tr>
<tr>
<td>LOW</td>
<td>15.58</td>
<td>8.09</td>
<td>15.20</td>
</tr>
<tr>
<td>OTH</td>
<td>8.30</td>
<td>9.07</td>
<td>7.49</td>
</tr>
</tbody>
</table>

Table 4: Mean proportions of ODC categories used in program summaries.

<table>
<thead>
<tr>
<th>Code</th>
<th>Object Description Category</th>
<th>Animation</th>
<th>Static</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>SD</td>
<td>Mean</td>
</tr>
<tr>
<td>PON</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PNO</td>
<td>3.13</td>
<td>8.84</td>
<td>1.14</td>
</tr>
<tr>
<td>PRR</td>
<td>5.55</td>
<td>11.87</td>
<td>11.14</td>
</tr>
<tr>
<td>PPD</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>DOM</td>
<td>88.54</td>
<td>12.63</td>
<td>80.74</td>
</tr>
<tr>
<td>IND</td>
<td>2.79</td>
<td>5.95</td>
<td>6.99</td>
</tr>
<tr>
<td>UNO</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 5: Participants’ evaluation of different characteristics of the two methods of visualization (scale 1-5); the best is 5.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Animation</th>
<th>Static</th>
</tr>
</thead>
<tbody>
<tr>
<td>Originality</td>
<td>3.88</td>
<td>0.64</td>
</tr>
<tr>
<td>Pleasure</td>
<td>3.75</td>
<td>0.89</td>
</tr>
<tr>
<td>Salience</td>
<td>4.38</td>
<td>0.74</td>
</tr>
<tr>
<td>Understandability</td>
<td>4.25</td>
<td>1.04</td>
</tr>
<tr>
<td>Usefulness</td>
<td>3.00</td>
<td>1.07</td>
</tr>
</tbody>
</table>
In order to study participants’ mental models of the studied programs, we used Good’s program summary analysis scheme [7]. For our analysis, we included the additional categories presented in [2]. The scheme consists of two classifications: one based on information types (IT) and the other based on object descriptions (ODC). The information types classification is used to code summary statements on the basis of the information types they contain. Table 3 contains the distribution of information type statements in each group. The object descriptions classification looks at the way in which objects are described. Table 4 contains the distribution of object description statements in each condition. Independent samples t-tests were carried out for both classifications. The only statistically significant difference between the groups was the the State-high category in information type classification \((t = 2.152, df = 14, p = 0.049)\).

We analysed the distribution of domain versus program information in participants’ program summaries further by using a similar strategy as Sajaniemi and Kuittinen [22]. We sorted program summaries into three types depending on the amount of domain versus program statements in object descriptions. Summaries with at least 67% domain statements (indirect and unclear statements excluded) were called domain-level summaries, summaries with at least 67% program and program only statements were classified as program-level summaries, and all others were called cross-referenced summaries because they had a more even distribution of domain and program information. The number of cross-referenced summaries was one in both groups.

Participants’ evaluation of the methods of visualization is presented in Table 5. Independent samples t-test was carried out, showing no statistically significant differences between the groups.

The correlations between the participants’ viewing times on different parts of the screen and other dependent variables—participants’ post-test score of variable role knowledge, and the program summary provided by the participants’—were analysed for both groups using the Pearson correlation coefficient. There were no significant differences.

4. DISCUSSION

In the experiment, we used two different versions of PlanAni program animator to visualize two short Java programs. The original version included animations of operations on variables, and displayed notifications for each program action. In the modified version, the operations simply replaced the old values of variables, and no notifications were displayed. With these two versions, we investigated how a person targets her visual attention, and what kind of a short-term mental model she constructs of a computer program and roles of variables. The experiment belongs to a series of experiments that will study in detail the short-term effects of visualizations.

The distribution of visual attention during the viewing provides an overview on what participants found interesting. Even though PlanAni is especially designed to visualize variables, the visual attention of the participants rested on variable area only a small proportion of time in both groups (14.96% in animation group and 10.79% in static group). It seems that the visualizations and animations are not by themselves sufficient to attract the visual attention of the participants to a large extent. Instead, the participants’ visual attention rested most of the time on the notifications, the code area and the IO area in the animation group (82.49%), and on the code area and the IO area in the static group (87.62%). Thus, even when participants were provided with rich pictorial information in the form of animated visualizations, they resorted heavily to the textual cues present on the screen.

In our previous experiment [15], we recorded visual attention between PlanAni version identical to the animation group and a textual Turbo Pascal debugger that displayed visualization in different location and with different size. In that experiment we found a significant difference in the proportional time spent on viewing the variable visualizations between the tools. These two results seem to suggest that differences in the location and size of variable visualization area have more influence on the distribution of visual attention than the presence or absence of an animation. The influence of pictorial versus textual presentation of variables may also have an influence on the distribution, and its effects should be studied in future.

In absolute time usage, the participants spent more time viewing the program with the animation version of PlanAni than with the static version. One explaining factor for the difference in the absolute viewing times are the notifications, which provided the animation group information on program actions, and enabled them to spend time viewing the information content on the screen at each step of the execution. The static group lacked the notifications, and they had to resort to other sources in order to gather information on program actions during the execution of the program.

To get a more detailed picture of how much animations and other screen events influence the distribution of visual attention, the video protocols were analyzed further qualitatively. Actions of the program animator were divided into four categories (highlighting of code when no other action takes place, display of a notification, animation of an operation, and print of output/request of input). The possible locations of viewer’s attention were divided into six categories (animations, other parts of the variable area, notifications, highlighted code, other parts of code, and IO area). Three one minute episodes of each participant’s protocol were analyzed (at the beginning, in the middle, at the end) by counting the number of occurrences of each action/location-pair. The count was increased every time participant’s visual attention located in one of the observed locations for several consecutive fixations. The data from this qualitative analysis is summarized in Table 6 for animations group, and in Table 7 for static group. The combinations of program action and locations of viewer’s attention that are impossible in practice are marked with N/A (not applicable). Counts that represent the default pairs, i.e., where the location of gaze is the currently visualized action, are marked in bold.

The general trend in Tables 6 and 7 shows that the participants focus most of their attention on the locations where program actions occur, regardless of the nature of the action. One exception for this are the IO actions, when participants share their attention evenly between the IO area and the highlighted code. During input, the IO area does not provide the name of the variable that will receive the entered data and it can be found in the program code only. During output, the animation version of PlanAni showed the expressions whose values were printed in the IO area whereas the static version did not. Tables 6 and 7 reveal that dur-
Table 6: Analysis of program actions and locations of visual attention for animations group.

<table>
<thead>
<tr>
<th>Program</th>
<th>Action</th>
<th>Location of Visual Attention</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Animation Variable Area</td>
<td>Notification</td>
</tr>
<tr>
<td>Animation</td>
<td>Beginning</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>Middle</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>End</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>31</td>
</tr>
</tbody>
</table>

| Notification | Beginning | 12 | 76 | 28 | 12 |  |
|          | Middle | 7 | 69 | 11 | 5 |  |
|          | End | 5 | 60 | 7 | 4 | 4 |
|          | Total | N/A | 24 | 205 | 46 | 21 | 4 |

| Highlighting | Beginning | 22 | 46 | 11 |  |
|          | Middle | 17 | 39 | 4 | 1 |
|          | End | 5 | 28 | 4 | 2 |
|          | Total | N/A | 44 | N/A | 113 | 19 | 3 |

| Input/Output | Beginning | 4 | 23 | 2 | 24 |
|          | Middle | 3 | 20 | 2 | 21 |
|          | End | 10 | 27 | 5 | 32 |
|          | Total | N/A | 17 | N/A | 70 | 9 | 77 |

Table 7: Analysis of program actions and locations of visual attention for static group.

<table>
<thead>
<tr>
<th>Program</th>
<th>Action</th>
<th>Location of Visual Attention</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Animation Variable Area</td>
<td>Notification</td>
</tr>
<tr>
<td>Highlighting</td>
<td>Beginning</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>Middle</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>End</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>N/A</td>
</tr>
</tbody>
</table>

| Input/Output | Beginning | 4 | 23 | 2 | 24 |
|          | Middle | 3 | 20 | 2 | 21 |
|          | End | 10 | 27 | 5 | 32 |
|          | Total | N/A | 17 | N/A | 70 | 9 | 77 |

ing IO, static group students directed their attention to the highlighted code more often than animation group students. Thus participants seemed to need information of the names of the variables involved in IO, and if this information was not in the IO area, they directed their visual attention to a place where the information could be found.

Tables 6 and 7 reveal a large difference between the groups in the amount of visual attention on the other parts of code during highlighting (19 versus 115 occasions). Other distinguishable difference between the groups is the difference in the amount of visual attention on the IO area. The difference is clear during the IO actions (41 versus 77 occasions), and also during other actions combined (9 versus 18 occasions). When animations and notification are absent, participants seem thus to resort more eagerly in browsing the code in general, and viewing the information in the IO area. This happens even during actions occurring on other parts of the screen.

There is also a general trend of concentrating more on the current action as the animation proceeds. At the beginning participants divide their visual attention more evenly among various locations but at the end they look at the on-going action. This could be a result of getting used to the animator interface—at the very beginning everything is new and interesting. The experimental sessions consisted, however, of two programs that were animated, and only the second one was taken into the analysis. Thus the larger division of attention at the beginning of animation is probably due to interest in the various parts of the new program rather than in the various parts of the animator.

A notable exception in the overall concentration on current actions is the static group’s large amount of visual attention on other code than the highlighted code at the end of animation. A similar effect cannot be found in animation group. It seems that due to the shorter absolute time that static group students spent with animation their understanding of the program was still developing and they had to study the program code more widely.

The mental representations of the studied programs did not, however, differ much. The only statistically signifi-
cant difference in the information type classification was in state-high category ($p = 0.049$). This suggests that the participants in the animation group gathered more high-level information on the viewed program than the static group. The statistical significance is, however, quite weak, and it is not supported by observations concerning other categories in the information type classifications. The object description classification did not reveal any significant differences between the groups.

Thus, no clear differences between the two groups in the short-term content of the mental models were found; neither could differences be found in an earlier experiment on short-term effects [15]. However, PlanAni has earlier been found to have positive long-term effects on programming skills and content of mental models [3, 21] when used for a longer period and with more active student engagement. There are several explanations for this. First, it could be that the differences in mental models of programs manifest themselves only after repeated occasions of viewing the visualizations. Second, it could be that the analysis method used does not catch small differences in the short-term effects of the visualizations. Finally, it may be that the style of engagement has been too simple in these experiments. The role of engagement in the viewing of visualizations has been found to play an important part in the beneficiality of visualizations [12] for learning. The use of style of engagement as a varied factor in future experiments may therefore provide interesting results for both the content of the mental models and the distribution of visual attention.

There were no statistically significant differences in the development of role knowledge between the groups. The animation group had a little higher mean score in the post-test but this was not statistically significant. When compared to an earlier study [23], from which the role tests were adopted, students’ performance on both the pre-test and the post-test was now much better. However, the number of roles in our experiment was limited to five, and the participants were shown an introductory video describing these five roles in the beginning of the experiment. This exhaustive introduction to the role concept can therefore be one possible reason for the small difference in the scores between the groups. In [23] the use of role images was found to improve the development of role knowledge compared to neutral control images. The difference in these two results seems to suggest that the role images, not the role animation, play central role in the development of role knowledge when PlanAni is used.

5. CONCLUSION

We have presented a model of the cognitive phenomena that take place during visualization viewing sessions. On the basis of the model, we are carrying out a series of experiments where the effects of visualization techniques are studied in detail. In the experiment described in the paper, we investigated how a person targets her visual attention, and how this affects to participant's mental models when a program is presented with two visualization tools that differed in the amount of animation. The visualization of programs and variables were the same but the animation displayed for assignment operations, comparisons, and input and output was removed from the other tool.

The results showed that the time participants spent viewing the variable area was quite low in both groups. Instead, the participants spent most of their time viewing highlighted code and notifications in animation group, and both highlighted and other code in static group. Thus, even when participants were provided with rich visual information in the form of animations, they resorted heavily to the textual cues present on the screen. As the animation proceeded, participants started to follow visual cues more closely—however, if the names of the variables involved in the current operation were not in the visually cued area, participants tended to search for them in the program code. In the absence of animation, participants were more eager to browse the code in general.

When the results of this experiment are analysed together with the results of earlier experiments on visualizing roles [15, 23], it seems that differences in the location and size of variable visualization area have more influence on the distribution of visual attention than the presence or absence of an animation. Also the role images, not the role animation, seem to play central role in the development of role knowledge when PlanAni is used.

These findings provide a challenge for the designers of visualizations, especially since the participants found the animated visualizations to be nevertheless understandable, pleasurable, and in a salient role for understanding the viewed programs.

6. ACKNOWLEDGMENTS

This work was supported by the Academy of Finland under grant number 206574.

7. REFERENCES


