Cite as: Osman, W. I., & Elmusharaf, M. M. (2014). Effectiveness of combining algorithm and program Animation: A case study with data structure course. *Issues in Informing Science and Information Technology, 11,* 155-168. Retrieved from http://iisit.org/Vol11/IISITv11p155-168Osman0478.pdf

Effectiveness of Combining Algorithm and Program Animation: A Case Study with Data Structure Course

Waleed Ibrahim Osman and Mudawi M. Elmusharaf
The National Ribat University, Khartoum, Sudan

walidibrahimosman@gmail.com **mudawi5@gmail.com**

Abstract

Data Structure is an important and mandatory course in Computer Science and Information Technology curricula. The topics of the course require detailed view of the algorithm behavior with the implementation part by a programming language. This paper introduces a visualized learning environment and assesses the effectiveness of using this system in order to enhance the education in computer science and information technology concerning data structure course. Whether the use of the system will produce remarkable enough affects to encourage the developers to enhance such systems. The system uses embedded visualizer rather than external visualizer to produce the animation. The significant result of the experiment indicates that combining Algorithm and Program Animation is an effective way to enhance students' performance.

Keywords: Algorithm animation, Program animation, Effectiveness of visualization, computer aided instruction, educational software.

Introduction

Learning Data Structure is not an easy task, according to several researches, students face problems in understanding an algorithm and its implemented piece of code, it remains a challenge in Computer Science education (Ben-Ari et al., 2011; Karavirta, 2007; Lahtinen, Ahoniemi, & Salo, 2007; Rajala, Laakso, Kaila, & Salako, 2008; Smith & Campus, 2000).

Numerous studies have addressed Algorithm Animation (AA) and Program Animation (PA) as separated parts. Regarding the use of Algorithm Animation in the teaching/learning environment, a number of authors Ben-Ari et al., 2011; Jarc, 1999) have reported mixed results. On the other hand, studies conducted to compare the effectiveness of Program Animation versus Algorithm Animation (Ben-Ari et al., 2011; Smith & Campus, 2000; Rajala et al., 2008) indicate the superiority of Program Animation over that of Algorithm Animation.

The study reported in this paper differs from most previous algorithm animation systems because it combines algorithm and program animation by providing the actual code script implemented in C and PASCAL languages of an algorithm, beside immediate runtime environment, which shows the current statement to be executed in addition to the control variables and their corresponding values during the runtime.

The system used the concept of Interesting Events (IE's) (Hundhausen, 2002; Kerren, & Stasko, 2002). In this approach the key points of the program are annotated with IE's by the developer, it injects the visualization mechanism inside the algorithm's script, instead of sending some information to external visualization engine; this insures that the animation of the program will be better, because it operates inside an actual running program.

Data Structure Courseware is a computerized system incorporates both PA and AA. It has been developed by the author as a teaching aid for the data structure course in order to enhance students' understanding of data structures and algorithms, such as sorting, searching, linked-lists, stack, queues, trees and binary search tree. The system has the capability to display data structures graphically on the computer screen, as well as allowing graphical manipulation of the data structure created. The students can observe the execution results and track the algorithm execution for selected algorithms, also in some cases the students can write some statements and observe the reflection, or the impact of that statement over the memory area.

Related Work

The courseware used in this study is an example of Computer Assisted Instruction (CAI) (Celikler & Aksan, 2011; Liao, 2007; Owusu, Monney, Appiah, & Wilmot, 2010; Ozmen, 2008), "it is the process by which written and visual information is presented in a logical sequence to a learner through a computer" as defined by Mahmood (2004). The concept of using computer in education is referred by several names such as: (1) Computer Aided Instruction (CAI) (Aqda, Hamidi, & Rahimi, 2011; Bintas & Camli, 2009; Desrochers & Gentry, 2004; Renshaw & Taylor, 2000); (2) Computer Aided Education (CAE) (Workman, 2004); (3) Computer Assisted Learning (CAL) (Cepni, Tas, & Kose, 2006; Gürbüz & Birgin, 2012; Liao, 2007; Ozmen, 2008); (4) Computer Based Education (CBE) (Renshaw & Taylor, 2000; See et al., 2010; Workman, 2004); (5) Computer Based Instruction (CBI) (Baturay & Bay, 2010; Means & Roschelle, 2010; Owusu et al., 2010; Powell, Aeby, & Carpenter-Aeby, 2003); (6) Computer Enriched Instruction (CEI) (Desrochers & Gentry, 2004; See et al., 2010); (7) Computer Managed Instruction (CMI) (Desrochers & Gentry, 2004; See et al., 2010); (8) Computer-Based Learning Environment (CBLE) (Künsting, Wirth, & Paas, 2011); and (9) Computer Assisted Teaching (CAT) (Gürbüz & Birgin, 2012). As mentioned before the system combined Program Animation and Algorithm Animation, so it is good to review the related studies about the effectiveness for each of them.

Program Animation

Program animation or Program visualization (Oechsle & Schmitt, 2002) can be defined as the use of computer graphics and animation techniques to visualize the behavior of an executing program (Noble & Groves 1991, p.1), program animation systems can be used by programming developers in order to debugging and understanding programs and also in teaching programming. Deihl (2007) has expressed the definition in short as Software visualization concerned with visualizing the structure, behavior, and evolution of software. Then he described structure as the static parts and relations of the system and behavior refer to the execution of program states where a program state contains both the current code and the data of the program.

The effectiveness of using program animation to improve students' learning outcomes has been examined through a fresh study done by Urquiza-Fuentes and Velázquez-Iturbide (2013). The study involved students of two computing degrees, management computing and computing systems, at Rey Juan Carlos University. The experiment concerned a Foundation of Programming Languages course offered to freshmen. The study reported that, students who used animations, either viewing or constructing them, got better grades than students who used the traditional approach which involved the text book and blackboard.

Another fresh study by Moons and De Backer (2013) clearly reported that the novel program visualization environment has a positive influence on the understanding of the students of a program containing a recursive method. The educational programming environment employed program visualization techniques to help teachers and students of an introductory programming course. The experiments showed that the environment can help with understanding the concepts of programming that most students consider very difficult.

A recent report (Ben-Ari et al., 2011) traces the instructional benefit of a program animation system called Jeliot. In this study the participants were high-school students studying introductory programming. The students were split into two groups, only one of which was exposed to animation. The group exposed to animation out-performed the group bereft of exposure to animation. Moreover, there appeared to be a learning advantage which persists over time: the group exposed to animation outperformed the other group in latter assignments too. This last finding could be explained as follows: while learning to use the animation tool does require time/effort, latter deployment of the tool, in similar learning situations, facilitates learning of advanced concepts.

A study of the effects of a system called ViLLE (Kaila, Rajala, Laakso, & Salakoski, 2010) has been conducted in a high school in Finland during 2010. The experiment considered a programming course. It reported that students who used the animation system ViLLE got significantly better results on the final exam.

An earlier empirical study held at the University of Turku "Finland" has reported that by using ViLLE, they obtained solid evidence that ViLLE enhances the learning of students with no prior programming experience substantially. The main objective of ViLLE was to provide an environment for students to study the execution of example programs, whether written by students themselves or prepared by the teacher (Rajala et al., 2008).

In 2000, Smith & Campus (in Smith & Campus, 200), using a visualization tool called Bradman, conducted an experiment to study the effects of deploying visualization to assist novices in learning to develop programming skills, using the programming language. This visualization system provides a framework for presenting a visual perspective of the changes occurring as the program’s commands are executed. The study for testing the efficacy of the methodology lasted three weeks and used 26 volunteers were split into two groups. The two groups were compared on pre- and post-test achievement scores. The study unequivocally indicated the beneficial effect of visualization in the learning of some of the programming concepts. Perhaps the more important conclusion of this study was that the student group that has used visualization was able to assimilated advanced topics with greater ease. Note that this last finding was also reported later by Ban-Ari, in 2011

In summarizing the reports discussed in this section there is evidence to conclude that incorporation of visualization into the teaching of programming facilitates learning. And that earlier adoption of visualization has an effect of imbuing students with insight that facilitates more advanced learning.

Algorithm Animation

Tudoreanu and Kraemer (2008) report from their recent study, in which their subjects were split into two groups: one group was provided with only text demonstration whereas the second group received a demonstration incorporating animation with text. The group receiving the animation augmented demonstration out-performed the other group.

An ethnographic study has been done by Hundhausen (2002). The study focused on having students construct and present their own visualizations. The report's finding was that conventional AV software can actually distract students from concentrating on activities and concepts relevant to an undergraduate algorithms course when used within the context of AV construction. He also reported that "experiments that manipulate learners' level of involvement have consistently yielded significant results, whereas experiments that manipulate representation have not".

Some of the earlier empirical studies on algorithm animation had elicited equivocal results, regarding benefit of incorporating animation as a tool to learning. Kehoe et al (2001) formulated some explanations about the failure of these earlier studies to find significant benefits. For the present, the explanation of note was the one that stated: "something in the design of the experiments is preventing participants from receiving the benefits, or in other words, the theory of how animations could help need to be re-examined". The authors conducted an observational study based on students using animation in a homework-style learning situation, (as opposed to an examination situation) to test the validity of their proposed explanation. The authors found that the group using animations out-performed the group that did not use animation.

Another empirical study has been done in 1998 at The George Washington University, it described a Web-based courseware that contains exercises that use algorithm animation and visualization, this courseware is intended for students in an introductory data structure course in Ada. The study measured whether the interactive feature included in the courseware influenced student performance and whether student learning styles were a factor. The study reported that no significant differences were found (Jarc & Feldman, 1998).

Embedded Animation Data Structure Courseware

The courseware was developed by the researcher to enhance the education process of computer science curricula, specially the data structure course. The system uses Program Animation and Algorithm Animation. The system consists of two units, Lecture unit and Lab. The lecture unit provides the text part of the course and it can be considered like a text book to introduce the concept of an algorithm, its steps, flowcharts, and implementation for it in PASCAL and C programming languages. This part is designed in HTML. Any browser can be used to view this unit. The lab unit is the cornerstone of the study. It provides the animation or visualization environment for each of the algorithms covered in the lecture unit. In sorting and searching algorithms, the array values are generated randomly by default, but in case of using this system by group there is a need to have the same values for all students, thus there is a subsystem that can be used to fill the arrays by predefined specified values, so each student will see the same array with the same values, and the instructor can continue demonstrating with identical lab values step by step. Normally the lab layout screen has four display areas whose description is given below:

Memory Mapping

This part dedicated to show the Data Values, the Diagrammatic depiction of the data structures, and Control Variables. For shapes depicting structures, the system employs the most commonly used figures. The data structures that can be depicted include: the Array, Lists, Circular Lists, Queues, Linked Lists, Stack, etc. Of course, the memory area also contains the control variables. As statements of the program are executed these control variables are also shown on the screen, with their names together with their current values. The embedded visualization system also depicts the System-Stack. The depiction of the System-Stack helps in diagnosing problems with recursive routines, by displaying all the modules still not exited.

Code display area

This area displays the part of the subroutine's currently running code, i.e. the statement currently under execution. This feature can provide a trace of statement execution.

System messages area

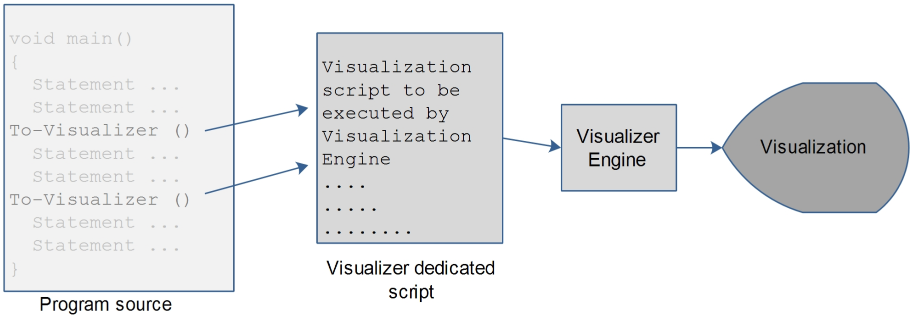
This area displays the messages generated during the execution, by the Embedded Animation Courseware System. The messages pertain to comments, tips, error messages, warning and hints that guide the user and provide useful information about the situation extant as well as guiding the user to possible actions.

User input area

This section is not included with every lab, it included only in labs that needs special input from the user such as function calls e.g. (Add(9); and DelTree();), linked lists statements (H->Next=P; and Q=NULL;). Function calls are just routines related to a programming language, so the student will feel like he/she is working with the compiler directly through the selected programming language.

Method of Visualization

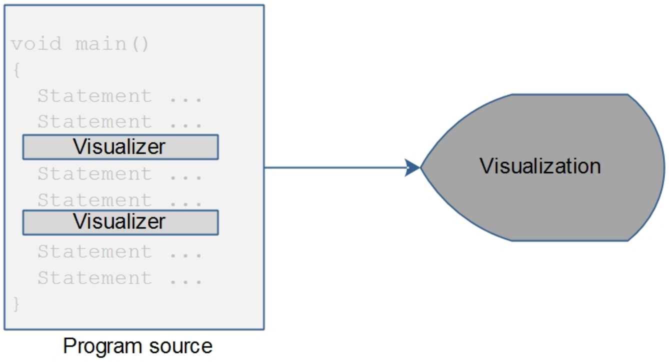
The normal method is to use an external visualizer system, so the program must send set of commands to that visualizer in order to animate the execution of the program and the data structure related with. The external visualization engine reads the commands from the generated file from the algorithm, and then starts to animate the algorithm after its actual program finished. This study is a report of the Embedded Animation Courseware System the author designed and implemented; the embedding strategy bypasses the necessity of the data-structure code having to send commands to the independent visualization system. The animation is performed within the program script and this method produces more reliable animations. The strategies of the normal method of visualization and the embedding method are illustrated through Figures 1 and 2.



```
void main()
{
  Statement ...
  Statement ...
To-Visualizer ()
  Statement ...
  Statement ...
To-Visualizer ()
  Statement ...
  Statement ...
}
```

```
Visualization
script to be
executed by
Visualization
Engine
....
.....
........
```

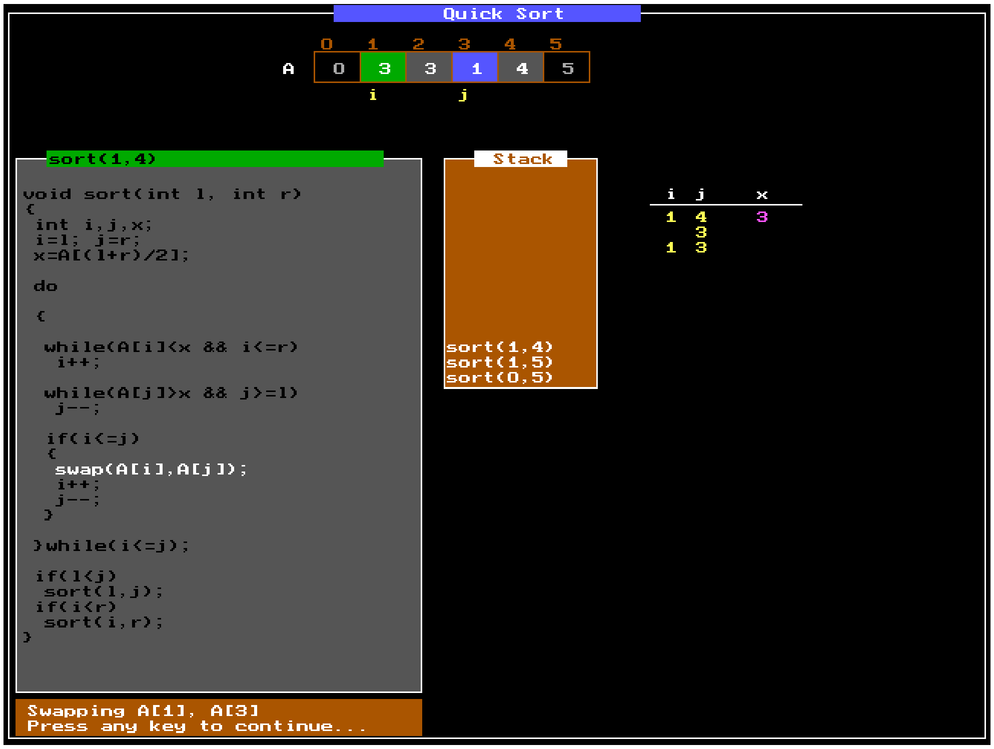
Figure 1 Normal Visualization method using external Visualizer



```
void main()
{
  Statement ...
  Statement ...
 Visualizer
  Statement ...
  Statement ...
 Visualizer
  Statement ...
  Statement ...
}
```

Figure 2 Advanced Visualization method using embedded Visualizer

For Graphical Systems, of which Animation Systems are a sub-genre, the color schema used for the display is of aesthetic as well as functional importance. From his study Ware (2013) recommended a list of 12 colors that can safely be used for displaying information on computer monitors. The colors mentioned in the list have widely agreed upon names and are easy to recognize visually because they are far apart in color-space. Ware's recommended colors are: Red, Green, Yellow, Blue, Black, White, Pink, Cyan, Gray, Orange, Brown and Purple. Presented in Figures 3 and 4 are screen-shots, from the Embedded Animation Courseware System using the colors recommended by Ware. Figure 3 displays the intermediate results of Quick-sort and Figure 4 displays the results for a Binary Search Tree.



```
void sort(int l, int r)
{
 int i,j,x;
 i=l; j=r;
 x=A[(l+r)/2];

 do

 {

  while(A[i]<x && i<=r)
   i++;

  while(A[j]>x && j>=l)
   j--;

  if(i<=j)
  {
   swap(A[i],A[j]);
   i++;
   j--;
  }

 }while(i<=j);

 if(l<j)
  sort(l,j);
 if(i<r)
  sort(i,r);
}
```

```
sort(1,4)
sort(1,5)
sort(0,5)
```

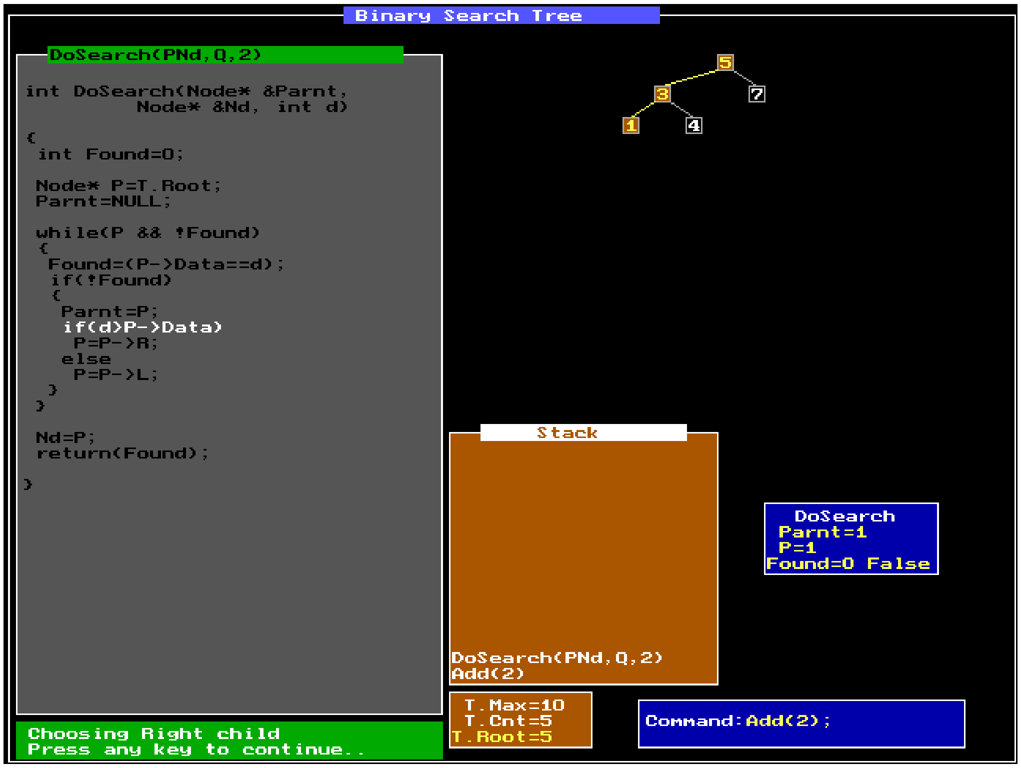
```
i  j    x
1  4    3
   3
1  3
```

```
Swapping A[1], A[3]
Press any key to continue...
```

Figure 3 Visualization System - Quick sort

The quick sort algorithm is demonstrated in Figure 3, provides the implementation in C++ at the left-hand side, indicates the current statement to be executed in white color, the array shape with values, indexes and special variables are showed on the top, the most important variables can be tracked with their in-progress values on the right-hand side, and the user will be guided and supplied by additional information through the system message area which appears at the bottom of the code area, in addition to the system stack. Through system stack, which is in brown color, the user has a chance to trace the recursion without missing the sequence, after the call of sort(1,4) finished, the algorithm will return to sort(1,5) in order to continue. The figure shows that the algorithm now is running sort(1,4) and from the code area the user can see that the current statement to execute is colored in white which is a function call of swap(A[i], A[j]), this function will be loaded, the system stack will grow, and the function code will be shown on the code area to start executing and showing its local variables.

The Binary Search Tree (BST) is demonstrated through Figures 4 and 5, additional parts were added, the new parts are directions and user input area, as showed in Figure 5. Directions is a list of possible commands or functions applied to the BST, and the user input is for accept the user command, so the student is free to call the suitable function. Through this lab, the student is able to observe the different aspects of performing BST's operations which includes (Add, Search, DelNode, DelTree and PrintTree). This lab also shows another blue bar at the right side of the stack as showed in Figure 4, this blue bar appears only to provide the user with some important information, like parameters' values and local function's variables with their in-time values, this information are helpful to maintain the overall understanding about the algorithm flow. Figure 4 shows that a tree with five nodes, and the user is adding new node, some nodes (7 and 4) are black colored, and the other nodes (5, 3 and 1) are in brown color, the brown nodes indicates the correct path to locate the new node which is (2), the path's lines were colored in yellow, the blue bar showing the current parent node is (1) and the new node (2) not existed, the green bar beneath the code area supplied the student with helpful information. Figure 5 shows the next step which will add the new node to be the right childe of node (1) as the green bar said.



DoSearch(PNd,Q,2)

```
int DoSearch(Node* &Parnt,
          Node* &Nd, int d)

{
 int Found=0;

 Node* P=T.Root;
 Parnt=NULL;

 while(P && !Found)
 {
  Found=(P->Data==d);
  if(!Found)
  {
   Parnt=P;
   if(d>P->Data)
    P=P->R;
   else
    P=P->L;
  }
 }

 Nd=P;
 return(Found);

}
```

```
DoSearch(PNd,Q,2)
Add(2)
```

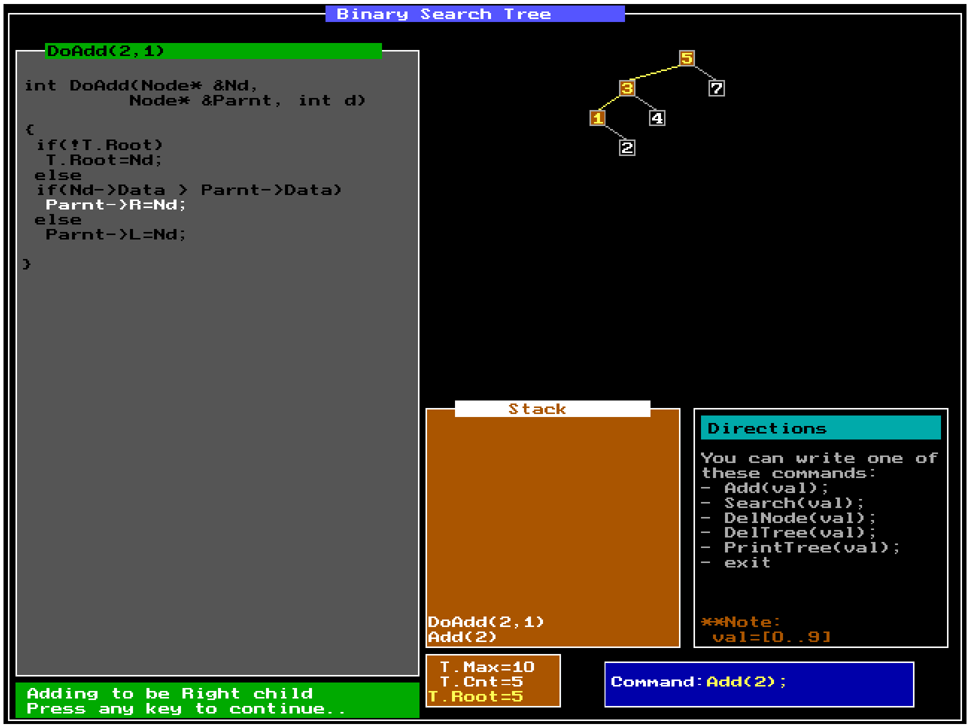
```
  DoSearch
 Parnt=1
 P=1
Found=0 False
```

```
 T.Max=10
 T.Cnt=5
T.Root=5
```

Command:Add(2);

Choosing Right child
Press any key to continue..

Figure 4 Visualization System - Binary Search Tree



```
DoAdd(2,1)

int DoAdd(Node* &Nd,
          Node* &Parnt, int d)

{
 if(!T.Root)
  T.Root=Nd;
 else
 if(Nd->Data > Parnt->Data)
  Parnt->R=Nd;
 else
  Parnt->L=Nd;

}
```

```
DoAdd(2,1)
Add(2)
```

```
You can write one of
these commands:
- Add(val);
- Search(val);
- DelNode(val);
- DelTree(val);
- PrintTree(val);
- exit

**Note:
 val=[0..9]
```

```
Adding to be Right child
Press any key to continue..
```

```
 T.Max=10
 T.Cnt=5
T.Root=5
```

```
Command:Add(2);
```

Figure 5 Visualization System - Binary Search Tree

Research Design

On the basis literature evidence as well as personal experience, the author formulated the conjectures:

1. Visualization of Algorithms helps students achieve greater understanding of the programming concepts.
2. A teaching instrument Combining Algorithm Animation with Program Animation would enhance student learning more than a System with External Visualization.

In this section the study describe the experiment, method, participants, and procedures.

Method

The explanatory method is the most suitable research method to use in this case, because the study will use experimental method through the posttest which is the final exam to assess the effectiveness of the tool. The experimental method permits researchers "not only to describe and predict but also to determine whether there is a cause-and-effect relationship between the variables of interest" as stated by Jackson (2011).

Posttest mechanism which specially designed to test the research hypothesis has been used by the study. The subjects of this study were the 4th semester students enrolled in department of Information Technology at Garden City College for Science & Technology during 2012 – 2013 academic year. Those students were learning a course in C++ and Data Structure course.

Tests were used as the measurement tool, although there are another three types of measurements, self-report, behavioral and physical (Jackson, 2009), tests are the best suitable tool to be used, because self-report is about doing questionnaire or interview, and the study did not intended to report the users' opinions, so this type of measurement was dropped off. Behavioral and physical measurements are now clearly unsuitable for the research, so they were also dropped off.

To be more precise, the study used t-test for Independent Groups instead of t-test for Correlated Groups, because the study has two groups extracted from a single population, the students enrolled in the 4th semester of Garden City College for Science & Technology, these two groups were compared by having one group received nothing more than the traditional way of delivering the course (the control group), and the second group received some level of the manipulated variable (the experimental group), which is the group used the visualization courseware beside the traditional method.

The statistical method that was used to measure the effectiveness is t-test. ANOVA is a famous statistical method, but it is not suitable in this case, because ANOVA is appropriate when the study has more than two groups to measure, but since this study has only two groups, t-test would be the best choice in hand to take as recommended by Jackson (2011).

Procedures

Generally there are three ways that algorithm animation can be used in an educational setting: (1) In the classroom lecture; (2) In laboratory exercises; (3) On demand, where the student can use the system in his/her private time at home.

The major concern of this research is to use the second way, which uses the system in laboratories. This system can also be used in classroom if the environment is well suited, like projectors and suitable white board dedicated for slide show with glare prevention capability.

The procedures are to wisely divide the selected students into two groups. In order to lessen the qualifications gap among them, and to reduce the differences between the two groups, the study sorted a list of participants according to their Cumulative Grad Point Average (CGPA), and then formulated two groups, one participant is selected to joined the first group, and the next participant on the list was joined to the second group, thus the participants were joined to the two groups interchangeably, Figure. 6 show how participants were distributed among the two groups.

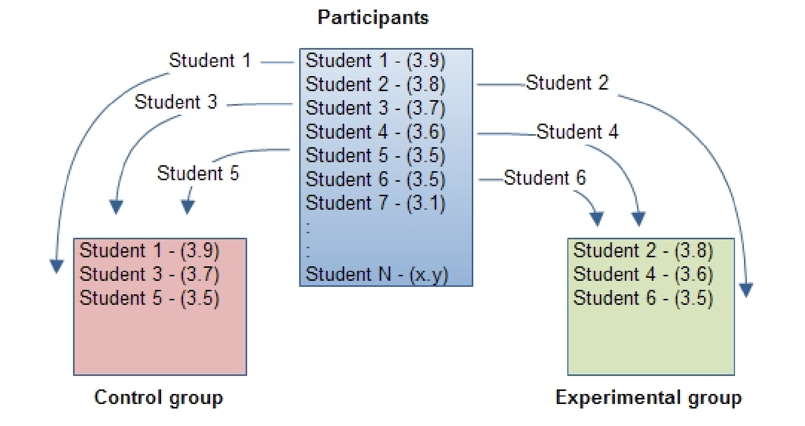


Figure 6 Distribution of participants among groups

The first group, which is the control group, will take some topics in the traditional way without the visualization system, which involves using text book and blackboard, while on the other hand the experimental group will take the same topics enhanced by the visualization system. After that, posttest has been taken and well interpreted in order to effectively answer the research hypothesis.

Results and Discussion

The participants were 40 students, divided into two groups, the Control group (C) involved 20 students, and the Experiment group (E) involved 20 students. Two students from E group were dropped out, because they had attended only two sessions out of eleven. Thus the E group now involved 18 students.

The dependent variable is the participants' scores on the final exam. These scores are listed in Table 1. At this point the research hypothesis can be formulated as:

Ha: Experiment group> Control group or µ2>µ1, which means that the study expect the E group will perform better than the C group. By running a t-test for independent group with the previous scores we can obtain the following results.

Notice that the mean performance of the experiment group (Visualization) = 44.05556 is better than that of control group (Traditional) = 33.35 as showed in Figure 7. However, we need to statistically analyze the data in order to determine whether the observed difference is statistically significant. The rest of the results showed in Table 2 indicated that the computed value of t = -2.5261, and by consulting the student's table to determine the critical value of t, with the degree of freedom = 36, and alpha=0.05. We find that for a one-tailed test, the critical value of t at .05 level is -1.684. Our t value is less than the critical value, thus the null hypothesis is rejected, and the alternative hypothesis that participants in the experiment group who used the visualization courseware will perform better in final exam than those students who did not is supported. A graph of the obtained t in relation to the t critical value is showed in Figure 8.

Table 1: Participants' final exam scores

Groups			
C		E	
58	35	60	40
54	32	59	40
52	32	57	38
50	30	56	36
47	24	55	34
39	16	51	30
39	14	48	28
39	14	46	28
36	10	44	-
36	10	43	-

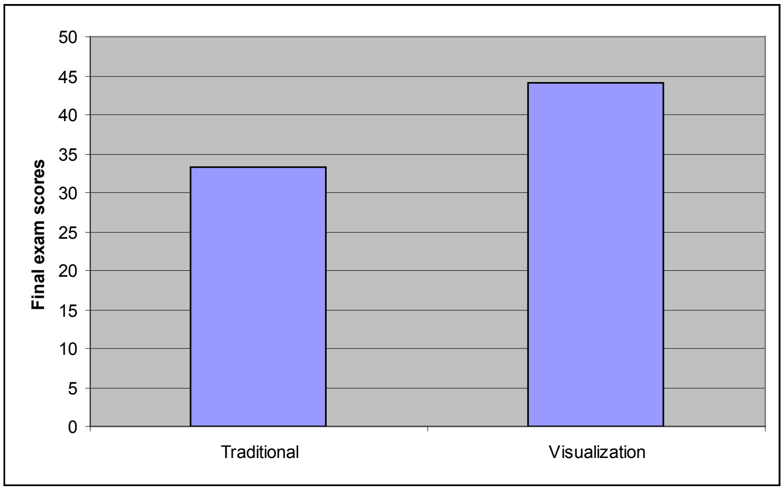


Figure 7 Mean performance of control group (Traditional) and experiment group (Visualization)

Table 2: One Tailed t-test Results Using STATA

Group	**Obs**	**Mean**	**Std. Err.**	**Std.Dev.**	**[95% Conf. Interval]**	
Control	20	33.35	3.324768	12.86881	26.39118	40.30882
Experiment	18	44.05556	2.50812	10.64105	38.76388	49.34723
Combined	38	38.42105	2.264712	13.96062	33.83231	43.0098
diff		-10.7056	4.237989			-21.411115
Diff = mean(C) – mean(E)					t = -2.5261	
Ho: diff = 0					degrees of freedom = 36	
Ha: diff < 0		Ha: diff != 0			Ha: diff > 0	
Pr(T < t) = 0.0080		Pr(T > t) = 0.0161			Pr(T > t) = 0.9920	

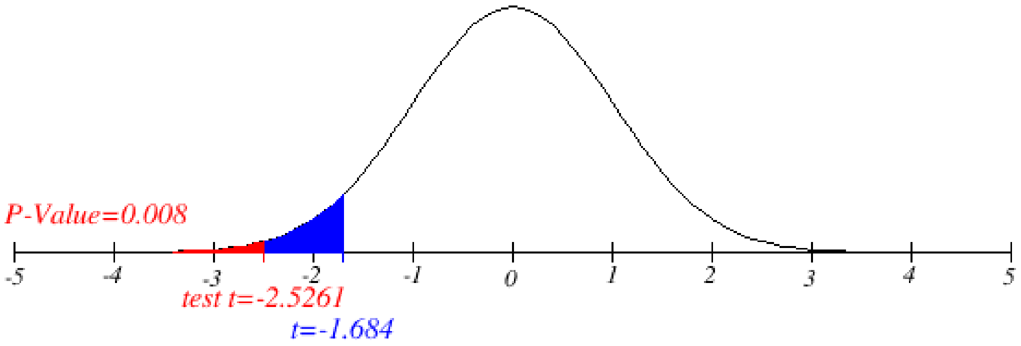


Figure 8 The obtained t-value (Red) in relation to the t critical value (Blue)

Conclusion

In this work, a new system with new mechanism has been made. The system differs from the related education system in several points. The implemented system combined both Algorithm Animation and Program Animation. The visualization engine was embedded with the program script, while the other systems uses external engine. The Visualization and use of the system doesn't involve rich interface, like Window-based Modal form, and using the Mouse, so it is clearly simple in order to avoid the user distraction. The system provided more simple interactivity by allowing the student to enter valid commands through the keyboard, rather than using the mouse selection and stop-pause fashioned, and to observe the affect of executing those statements live.

The impact of using the system has been evaluated and tested. The experiment that conducted provided strong evidence that this system is beneficial in teaching Data Structure. Based on the results provided, and the significance reported, we conclude that using Data Structure courseware enhanced the students' understanding which positively reflected on their results of the final exam.

The gained experience during developing and using the data structures courseware together with the experiments conducted have led to believe that there is a need for further work in several areas. The well trained instructors to effectively use the visualization systems are needed. The second is exploring how audio, specifically narration, could be used to improve the understandability of animations and project more easiness on the system, but it must be done carefully to avoid distracting the students. Next, trying to implement such system in Web platform would be good in publishing the system and gaining wider audience with ease of access. Finally, finding suitable ways and trained developers to reduce the cost of developing courseware containing animations is needed to make this process easier and less time consuming.

References

Aqda, M.F., Hamidi, F., & Rahimi, M. (2011). The comparative effect of computer-aided instruction and traditional teaching on student’s creativity in math classes. *Procedia Computer Science*, *3*, 266-270.

Baturay, M., & Bay, O. (2010). The effects of problem-based learning on the classroom community perceptions and achievement of web-based education students. *Computers & Education*, *55* (1), 43-52.

Ben-Ari, M., Bednarik, R., Levy, R., Ebel, G., Moreno, A., Myller, N., & Sutinen, E. (2011). A decade of research and development on program animation: The Jeliot experience. *Journal of Visual Languages & Computing*, *22*(5), 375-384.

Bintas, J., & Camli, H. (2009). The effect of computer aided instruction on students’ success in solving LCM and GCF problems. *Procedia - Social and Behavioral Sciences*, *1*(1), 277-280.

Celikler, D. & Aksan, Z. (2011). The effect of computer assisted instruction in teaching ionic compounds on pre-service elementary science teachers’ academic achievement and permanent learning. Procedia - *Social and Behavioral Sciences*, *28*, 547-552.

Cepni, S., Tas, E., & Kose, S. (2006). The effects of computer-assisted material on students’ cognitive levels, misconceptions and attitudes towards science. *Computers & Education*, *46*(2), 192-205.

Desrochers, M. N., & Gentry, G.D. (2004). Effective use of computers in instruction. In J. D. Moran & R. W. Malott (Eds.), *Evidence-based educational methods* (pp. 127-141). San Diego: Academic Press.

Diehl, S. (2007). *Software visualization: Visualizing the structure, behaviour, and evolution of software*. Berlin Heidelberg: Springer-Verlag.

Gürbüz, R., & Birgin, O. (2012). The effect of computer-assisted teaching on remedying misconceptions: The case of the subject “probability”. *Computers & Education*, *58*(3), 931-941.

Hundhausen, C. D. (2002). Integrating algorithm visualization technology into an undergraduate algorithms course: Ethnographic studies of a social constructivist approach. *Computers & Education*, *39*(3), 237-260.

Jackson, S.L. (2009). *Research Methods and Statistics: A Critical Thinking Approach*. (3rd ed.). Belmont USA: Wadsworth.

Jackson, S. L. (2011). *Research methods: A modular approach*. (2nd ed.). Belmont USA: Wadsworth.

Jarc, D. J. (1999). *Assessing the benefits of interactivity and the influence of learning styles on the effectiveness of algorithm animation using web-based data structures courseware*. (Doctoral dissertation, The George Washington University). Retrieved from http://students.seas.gwu.edu/~idsv/

Jarc, D. J., & Feldman, M. B. (1998). An empirical study of Web-based algorithm animation courseware in an Ada data structure course. *Proceedings of the annual ACM SIGAda international conference on Ada,* 1998, pp. 68-74.

Kaila, E., Rajala, T., Laakso, M., & Salakoski, T. (2010). Effects of course-long use of a program visualization tool. In T. Clear & J. Hamer (Eds.) *Proceedings of the 12th Australasian Computing Education Conference*. Brisbane, Australia: Australian Computer Society, pp. 97-106.

Karavirta, V. (2007). Integrating algorithm visualization systems. *Electronic Notes in Theoretical Computer Science*, *178*, 79-87.

Kehoe, C., Stasko, J., & Taylor, A. (2001). Rethinking the evaluation of algorithm animations as learning aids: An observational study. *International Journal of Human-Computer Studies*, *54*(2), 265-284.

Kerren, A., & Stasko, J. T. (2002). Chapter 1: Algorithm animation. In Diehl, S. (Ed.), *Software visualization* (pp. 1-15). Heidelberg: Springer Berlin.

Künsting, J., Wirth, J., & Paas, F. (2011). The goal specificity effect on strategy use and instructional efficiency during computer-based scientific discovery learning. *Computers & Education*, *56*(3), 668-679.

Lahtinen, E., Ahoniemi, T., & Salo, A. (2007). Effectiveness of integrating program visualizations to a programming course. *Proceedings of the 7th Baltic Sea Conference on Computing Education Research (Koli Calling 2007)*. Koli National Park, Finland: ACS, pp. 195-198.

Liao, Y. (2007). Effects of computer-assisted instruction on students' achievement in Taiwan: A meta-analysis. *Computers & Education*, *48*(2), 216-233.

Mahmood, M. (2004). *A comparison of traditional method and computer assisted instruction on students achievement in general science*. (Doctoral dissertation, University of the Punjab). Retrieved from http://prr.hec.gov.pk/Chapters/1405-0.pdf

Means, B., & Roschelle, J. (2010). An overview of technology and learning. In P. Peterson, E., Baker, & B. Mcgaw (Eds.), *International Encyclopedia of Education*. (3rd ed.) (pp. 1-10). Oxford: Elsevier.

Moons, J., & De Backer, C. (2013). The design and pilot evaluation of an interactive learning environment for introductory programming influenced by cognitive load theory and constructivism. *Computers & Education*, *60*(1), 368-384.

Noble, R. J., & Groves, L. J. (1991). Tarraingim - A program animation environment. *Proceedings of the 12th New Zealand Computer Conference, Dunedin*, 14-16.

Oechsle, R., & Schmitt, T. (2002). JAVAVIS: Automatic program visualization with object and sequence diagrams using the java debug interface (JDI). *Lecture Notes in Computer Science*, *2269*, 176-190.

Owusu, K. A., Monney, K. A., Appiah, J. Y., & Wilmot, E. M. (2010). Effects of computer-assisted instruction on performance of senior high school biology students in Ghana. *Computers & Education*, *55*(2), 904-910.

Ozmen, H. (2008). The influence of computer-assisted instruction on students' conceptual understanding of chemical bonding and attitude toward chemistry: A case for Turkey. *Computers & Education*, *51*(1), 423-438.

Powell, J. V., Aeby, V. G., & Carpenter-Aeby, T. (2003). A comparison of student outcomes with and without teacher facilitated computer-based instruction. *Computers & Education*, *40*(2), 183-191.

Rajala, T., Laakso, M., Kaila, E., & Salako, T. (2008). Effectiveness of program visualization: A case study with the ViLLE tool. *Journal of Information Technology Education: Innovations in Practice*, *7*, 15-32. Retrieved from http://www.jite.org/documents/Vol7/JITEv7IIP015-032Rajala394.pdf

Renshaw, C. E., & Taylor, H. A. (2000). The educational effectiveness of computer-based instruction. *Computers & Geosciences*, *26*(6), 677-682.

See, L., Huang, Y., Chang, Y., Chiu, Y., Chen, Y., & Napper, V. S. (2010). Computer-enriched instruction (CEI) is better for preview material instead of review material: An example of a biostatistics chapter, the central limit theorem. *Computers & Education*, *55*(1), 285-291.

Smith, P. A., & Campus, M. H. (2000). The efficacy of a low-level program visualization tool for teaching programming concepts to novice C programmers. *Journal of Education Computing Research*, *22*(2), 187-215.

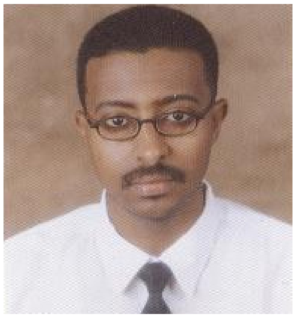
Tudoreanu, M. E. & Kraemer, E. (2008). Balanced cognitive load significantly improves the effectiveness of algorithm animation as a problem-solving tool. *Journal of Visual Languages & Computing*, *19*(5), 598-616.

Urquiza-Fuentes, J., & Velázquez-Iturbide, J. Á. (2013). Toward the effective use of educational program animations: The roles of student's engagement and topic complexity. *Computers & Education*, *67*, 178-192.

Ware, C. (2013). Color. In C. Ware, *Information visualization* (3rd ed.) (pp. 95-138). Boston: Morgan Kaufmann Publishers.

Workman, M. (2004). Performance and perceived effectiveness in computer-based and computer-aided education: do cognitive styles make a difference? *Computers in Human Behavior*, *20*(4), 517-534.

Biographies



Waleed Ibrahim Osman received the B.Sc. in Computer Science from University of Science & Technology, Omdurman, Sudan in 2003, and the M.Sc. in Computer Science from University of Gezira, Wad Medani, Sudan in 2006. Currently he is a PhD. Student in Computer Science at the National Ribat University, Khartoum, Sudan, and is a lecturer in college of Computer Studies, the National Ribat University. His research interests include visualization, algorithms and e-learning.

.

Dr. Mudawi M. Elmusharaf received the B.Sc. in Education from University of Khartoum, Sudan in 1974, and the M.Sc. in Educational Technology from University of Georgia, USA in 1985, PhD. In Computer based education from Southern Illinois University, USA in 1989. Currently he is an Associate professor in the College of Computer Studies, the National Ribat University. His research interests include information security, e-crime, e-learning, and educational technology.