International Journal of Innovation, Management and Technology, Vol. 2, No. 1, February, 2011
ISSN: 2010-0248

Interactivity in Data Structures and Algorithm Courses with SKA for RBTs

Kadian Davis and Dr. A.G. Hamilton Taylor

***Abstract*—Developments in multimedia and e-learning software have prompted many academics to express interest in using educational software. In our research, we have designed an extension of an algorithm visualization (AV) system, with the goal of enhancing the learning of students within a data structures and algorithms course. We were primarily concerned with designing an AV system to meet its users' needs. Therefore, we used a user-centred design approach to design the extension of the AV system. This journal reports on the impact of using multimedia technologies within a data structures and algorithms context. A highly interactive red-black tree (RBT) AV based on SKA (Support Kit for Animation) was designed and evaluated with a group of undergraduate students. Student performance was evaluated through a pre- and post-test and the results show a significantly improved post-test performance by the students. Based on the results of this study, the authors suggest the feasibility of using multimedia in e-learning.**

***Index Terms*—algorithm visualization, data structures and algorithms, red-black trees, user-centered design.**

I. Introduction

Computer Science involves the study of computers covering both aspects of hardware and software design. However, there are many areas of computer science that students often regard as challenging [1]. One of these areas is data structures and algorithms (DS&A), which is a traditional course included in computer science majors. A data structure is meant to be an organization or structuring for a collection of data items [2]. They are often depicted as diagrams or implemented as program code, and consequently, students often experience difficulties in understanding how they relate to each other. As a consequence, a number of system developers such as those in [3]–[5] conjectured that algorithm visualizations (AVs) would be of high pedagogical value when applied to the teaching of DS&A.

Multimedia in e-learning can be displayed as text, sound, still graphic images, animation or video. Furthermore, multimedia technologies can be integrated to create and design interactive e-learning tools. Software visualization is defined in [6] as: "the use of the crafts of typography, graphic design, animation, and cinematography with modern HCI (Human Computer Interaction) technology to facilitate both the human understanding and effective use of computer software." Algorithm Visualization (AV) is a sub-field of software visualization, which aims to narrow the visual cognitive gap by using computer graphics to portray the data structures as diagrams, and animation to show how the algorithms modify these diagrams [1], [7].

With the advent of computer graphics, high expectations were created in the computer science teaching community [8]. It was thought that educational software would increase learning flexibility in comparison to the use of static media such as overhead projectors, and white-boards while teaching DS&As. Limitations of AV system development in earlier years led to the production of movie-type animations [3] that were not interactive. As a result, their usefulness was limited in active learning scenarios. Following the development of movie-type animations, a number of interactive systems were developed [4]–[6], which facilitated sound, color, input data, control of the animation speed and multiple views.

Since the inception of AV, a plethora of studies on the use of visualizations have shown mixed results. Studies which demonstrate that AV software can be helpful are seen in [9], [10] while studies which illustrate an insignificant or questionable benefit from using AV software are illustrated in [9], [11].

Due to the sometimes inconclusive results of the effectiveness of AVs in university-level education, a hypothesis was proposed and investigated that learning might be enhanced with the use of well designed interactive AV software.

The rest of this journal is organized as follows. Section II describes the motivation for algorithm visualization. The challenges of and design recommendations for multimedia technologies in DS&A courses are discussed in Section III. The study is described in Section IV. Section V presents future work and Section VI concludes the paper.

II. Motivation: Why Algorithm Visualization?

The surveys conducted by Naps et al. [12] suggested that visualizations could have a positive impact on learning. Khuri lays emphasis on interactive AVs providing new opportunities for instruction [13]. In addition, Khuri states that AVs can enhance communication between students and lecturers. This is supported by his argument that student-lecturer interaction is minimal due to the frequent "one-directional" process depicted in conventional classrooms [13]. In addition, Khuri discussed that interaction is limited to oral discussions and it is very rare that students

Manuscript submitted on January 14, 2011.

Kadian Davis, The Department of Computing. The University of the West Indies. Mona, Jamaica, (email: kadian.davis@gmail.com).

A. G. Hamilton-Taylor, Dr., The Department of Computing. The University of the West Indies. Mona, Jamaica, (e-mail: ashley.taylor@uwimona.edu.jm).

have the opportunity to explore a concept in other dimensions during class time.

The following points represent the views regarding the use, benefits, and applicability of AVs from selected researchers.

1) They can assist students as they learn about algorithm concepts in a computer science course (Hundhausen et al. 2002) [9]. Consequently, students could be better helped to conceptualize specific algorithm operations as shown by the AV system.
2) Kraemer argued that the visualization can be helpful in allowing the user to understand the concurrency of the program and in managing the large amounts of objects, which are liable to be found in both parallel and distributed algorithms [14]. As a result, apposite exhibits of synchronized programs can help viewers to perceive the performance and correctness problems that result from unanticipated interactions between processes.
3) Byrne et al. stated that AVs might influence the learner to make and test predictions of what will happen for each step of the algorithm [15]. This may be advantageous in allowing the learner to understand algorithmic concepts better than passive learning. However, this claim does not always guarantee success.
4) Yeh [16] discussed that some textbook diagrams fail to present the learning material in a manner that stimulates the learning interest of students. Therefore, when compared to textual or static representations, motion images can be efficient in providing a more accurate graphical view of the steps involved in the algorithm. In addition, animations can be more enjoyable than explanations that are static or textual and consequently can be an intrinsic learning motivator.

Therefore, the factors mentioned above suggest that properly designed AVs would be beneficial to both lecturers and students. Thus, their adoption should be seriously considered.

III. Challenges And Design Recommendations For Multimedia Technologies In Ds&A Courses

Specifically, the marriage between multimedia technologies and DS&A initiated the development of interactive AV software for DS&A courses. Though students and instructors have sparked interest in AV, some academics are still reluctant to employ AVs in instruction. This could be as a result of perceived ineffectiveness of AVs. Predominantly, the ineffectiveness of an AV may be associated with the overhead faced by the educators in making the visualization meaningful to students [17]. The pre-conference survey by Naps et al. [12] revealed the factors that made respondents or their colleagues reluctant or unable to use dynamic visualizations. Some of the reasons for reluctance to adopt are shown below:

- 93%: time required to search for good examples
- 90%: time it takes to learn the new tools
- 90%: time it takes to develop visualizations
- 83%: lack of effective development tools
- 79%: time it takes to adapt visualizations to teaching approach and/or course content
- 69%: lack of effective and reliable software

Although there are several questions and reasons regarding the success of AV adoption, instructors and students alike must pay significant attention to the factors that contribute to the educational effectiveness of AVs. Thereafter, they can make their evaluations of the educational effectiveness of the AVs. Recommendations for good AV design are described in [12], [16], [18] some of which are listed below:

- AVs should be accompanied with comprehensive motivational instructions. In order to stimulate this instruction, the animation display should be augmented by textual explanations of the ongoing operations.
- AVs should be consistent with the material presented in the textbook in order to provide clarity to students working with AVs.
- AV designers should perform user testing on the AV before employing it in a classroom setting. The feedback from students can be valuable in improving the instructional quality of the animation.
- The animation system should support highlighting and marking.
- AVs should support user input of data.

Furthermore, we propose the following additional recommendations, which could be applied to multimedia in e-learning domains.

- Multimedia teaching tools should support attractive user interface elements. The animated drawings should be clear and legible. The visualization should be attention grabbing but not distracting and should support aesthetic features such as the correct use of colors.
- The user interface should focus the user's attention. Multimedia tool designers should avoid too many concurrent actions in visualization software (i.e. too many things happening at the same time). This should reduce the visual clutter.
- All operations of the multimedia software should be fully functional. Therefore, when controls such as buttons are used they should work.

We believe that designers of multimedia software should pay close attention to the mentioned practices in order to ensure that the system designed will support the users' needs. Pragmatically, if the set of design recommendations described above are considered in designing multimedia teaching tools, then usability concerns can be significantly reduced.

IV. The Study

A. Background

Cormen et al. (1990) [19] describe RBTs as follows: A red-black tree is a binary search tree with one extra bit of storage per node: its color, which can be either RED or BLACK. By constraining the way nodes can be colored on any path from the root to a leaf, red-black trees ensure that no such path is more than twice as long as any other, so that the tree is approximately balanced." This guarantees that thebasic operations take place in O(lg n) time in the worst case. However, this topic has a reputation for being difficult

to learn. An informal survey administered at the University of the West Indies, Mona (UWI) in 2008 revealed that approximately seventy percent (70%) of the thirty DS&A students identified RBTs as being the most difficult topic in the DS&A course, and therefore required an AV that supported this data structure. In particular, one student stated, "Of all the sorting algorithms, red-black trees is the hardest and an animation tool would surely help." Similarly, both the head of department and lecturer of the DS&A course at UWI desired an AV that supported RBTs.

In this research, we conducted an expert evaluation (informal cognitive walk-through) of six existing RBT AVs namely; Java Models Red-black, Red-black Tree Demonstration, Red-black Tree Simulation, Kovac's Red-black Tree, Trackla2 and JHAVE. The results of our evaluation were partly sourced from http://wiki.algoviz.org/AlgovizWiki/RedBlackTrees. Our findings suggested that only one of the systems examined (Trackla2) partially supported manual operations. It is believed that manual operations are necessary to ensure active participation with the AV software and consequently, improve the learning of DS&A concepts. This conjecture is supported by Papert's assertion of Constructionism discussed in [20]. Therefore, when students manually construct RBTs using AV software then learning might be improved. An additional limitation of the AV systems examined was their inability to support highlighting and marking of nodes. Also, a significant drawback in the six AV systems examined was their inability to clarify why a particular step was needed. We believe that an explanation can be useful in increasing the understanding and learning of the necessity for the algorithmic steps. Therefore, an explanation facility was included in the SKA RBT extension. Also, the inclusion of examples in the AV software could facilitate student-learning and problem-solving for algorithmic problems.

SKA is a multimedia tool that combines an interactive data structure diagram manipulation environment, an algorithm animation system and a visual data structure library [21]. However, the initial version of the SKA visual data structure library (graphs, binary trees and arrays) did not support RBTs. SKA supports operations for the construction and manipulation of data structure diagrams. SKA makes it easy to animate an algorithm (written in Java), while controlling execution via a pseudocode version of the algorithm. SKA also includes execution pacing control.

Importantly, SKA provides an extensible framework for the addition of new data structures. Therefore, RBTs were added, and additional features were introduced to the SKA environment in order to support scaffolding-type feedback while constructing RBTs. SKA was designed to facilitate discussion and exploration with students [21]. In addition, SKA can be used outside the class-room teaching environment to enhance self-directed learning of algorithms.

B. SKA For RBTs Design

SKA for RBTs was designed to address the usability issues that were found in the expert evaluation. We adopted

Beyer and Holtzblatt's [22] contextual design (CD) approach, which is a user-centered approach to software design, to drive the design of the extension of SKA. Basically, CD is an iterative process which collects multiple customer-centered techniques into an integrated design process [22]. We believe that a user-centered approach to multimedia software design will result in multimedia teaching tools that are far more relevant to the needs of both instructors and students. As a result, our ethnographic inquiry consisted of interviewing, observing and video-taping stakeholders (lectures and students) as they carried out their tasks within the DS&A environment. During the observational session, notes were taken on the artifacts used to perform tasks within a DS&A context. Multiple artifacts such as: the whiteboard, markers, examples, lecture notes, CS20A manual, node shapes, books, pens, pencils, laptop and desktop computers were identified during the instructional study. However, this manuscript shows only the main artifact model adapted from contextual design, which describes how the artifacts were used during a DS&A lecture is demonstrated in Figure 1. As shown in the artifact model, the major objectives of the department head (DS&A lecturer) were teaching the properties of RBTs and rules for insertion. As a result, to support the context of use during this phase of the study, the SKA extension for RBTs was redesigned to support insertion in RBTs while using a step by step approach to problem solving.

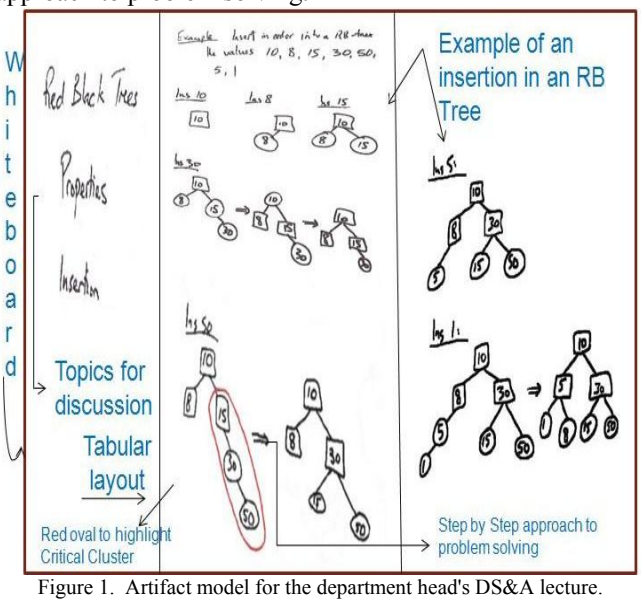


Figure 1. Artifact model for the department head's DS&A lecture.

In essence, the results of our ethnographic inquiry portrayed the instructional artifacts and their uses in DS&A lectures, which were instrumental in suggesting areas for potential technological task support in SKA for RBTs. In addition, we were informed of the learning difficulties that the students faced within the DS&A context. In an interview conducted with the head of the computing department at UWI he stated that: "students are usually confused at first about the reasons for the rules of RBTs and how they provide self-balancing behavior; but this is probably a natural part of the learning process. In order to understand RBTs, one needs to first become familiar with the rules, and then the reasons why they lead to a balanced tree. The former is sometimes difficult to grasp because the context may seem abstract (i.e. trying to remember that every node has one of 2 colors, and that the black path length to every leaf must be the same, and

so on will appear arbitrary to the newcomer). The latter is difficult to grasp because it requires a bit of mathematical reasoning to see how the rules lead to a tree that is always balanced.

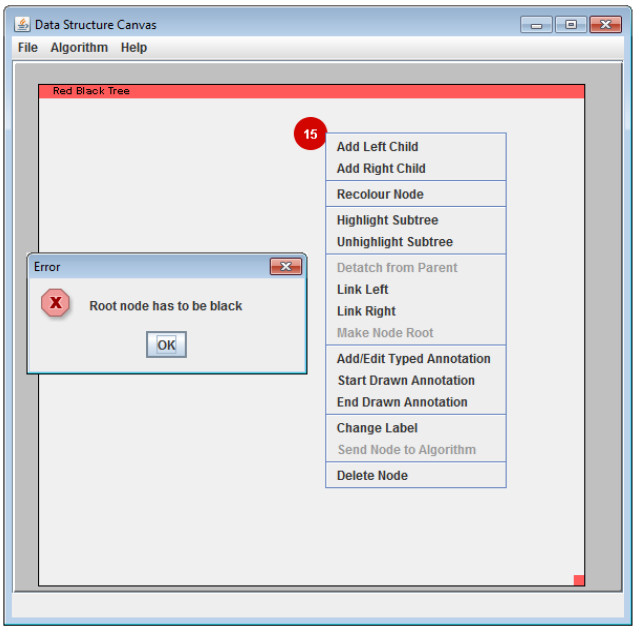


Figure 2. The user right-clicks the root node to invoke the menu, then tries to add a right or left child. Error Message hinting the user must recolor the root node before adding descendants. This is useful if the user is unfamiliar with the rules of insertion in red-black trees.

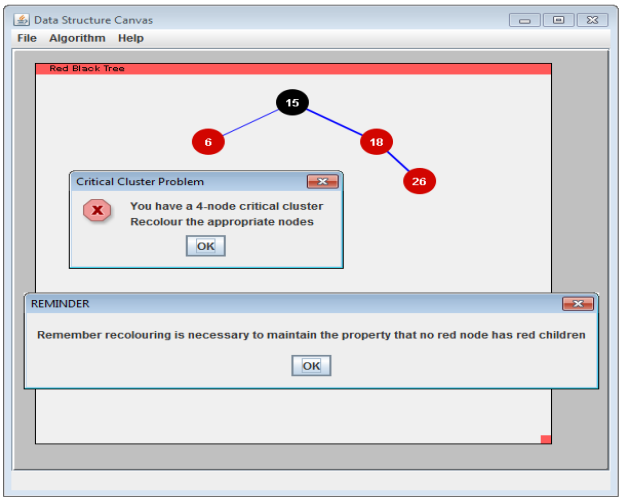


Figure 3. Since 26 is greater than root node value 15, and is greater than 18, it is inserted as 18's right child. Upon insertion of the value 26, the user is informed of a four node (4-Node) critical cluster problem. He or she is instructed to recolor the appropriate nodes and is further reminded of the reasons for recoloring.

As a result, SKA for RBTs was extended to include a concrete rule-list explaining the reasons for the rules and how they provide a self-balancing behavior. Furthermore, it is equipped with clear help instructions and RBT examples ranging from simple to difficult RBTs. We decided to improve the interactivity of SKA for RBTs by providing reminder facilities and explanations on the significance of each algorithmic step. Our extension of SKA supported on-the-fly checks for validation and the system was redesigned to also validate the solution upon completion. In addition, SKA for RBTs facilitates manual operations by allowing its users to construct, manipulate, highlight, and annotate RBT data structure diagrams.

As a result, we believe that the SKA RBT extension will be helpful in enhancing the learning of RBTs. Figures 2-4 show some interactive features of the SKA RBT extension.

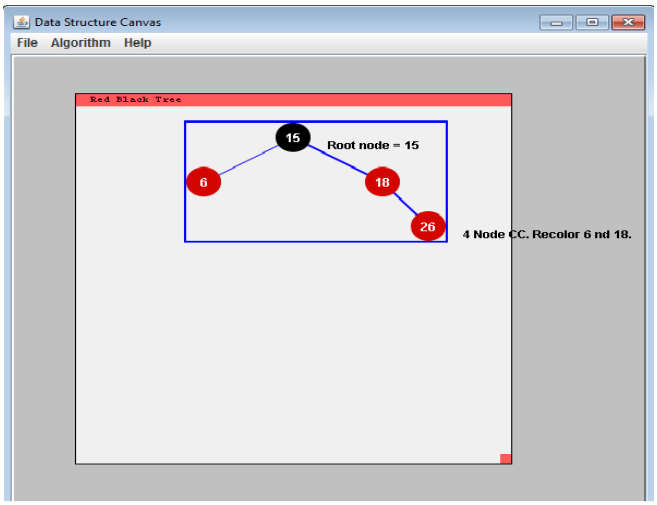


Figure 4. The user selects Highlight Subtree, and highlights the critical cluster. The user then selects Add/Edit Typed Annotation to annotate his RBT data structure diagram.

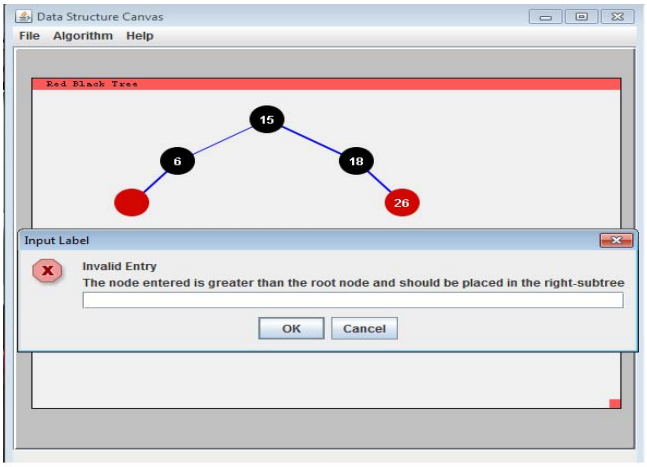


Figure 5. The user selects Add Left Child, and subsequently enters 14 as the left node value. The user is prompted with message stating that the inserted value should be placed in right sub-tree. The user can cancel his/her addition of a left node by clicking Cancel. Otherwise, the user is able to enter a value that is less than the parent node.

C. *Algorithm Visualization Software Testing*

Testing of the implemented AV software was conducted between two groups (A and B) with group A comprising of mature students (high school information technology teachers) registered to the summer offering of the DS&A undergraduate course. Group B consisted of students about to enter the second year of the undergraduate Computer Science program who had never taken the DS&A course, with the exception of two mature students who arrived late for the summer DS&A course. Note that the latter group only saw a video of the lecture on the BSTs and RBTs (video of the lecture presented to group A). In this study, the video of the lecture was used to enable teaching of students who were not physically present at the lecture session.

The procedure for experimentation was as follows:

- Both groups read the rule-list on BSTs and RBTs for fifteen minutes.
- The rule-list was taken away and they sat a pre-test for thirty minutes.
- Then an interactive learning exercise was performed using SKA for RBTs with the rule-list as a reference tool.
- The user experiences while using the software were observed.
- Subsequently, the reference tools (SKA RBT AV and rule-list) were removed and a post-test given for thirty minutes.
- After completing the post-test, both groups were asked to complete questionnaires regarding their experiences while using the SKA RBT AV and their recommendations for further system development.
- The *student two tailed t-test* was applied to the pre- and post-test results of both groups to evaluate the statistical significance of the results of the study.

D. The Findings

Participants of group B were more receptive to the SKA extension for RBTs when compared to group A participants who became frustrated while using SKA. However, after consulting the SKA help file the majority of group A's problems were resolved.

When the participants were questioned on their thoughts on the overall performance of SKA, 58% said it was somewhat efficient, 16% said it was efficient, while 25% said it was not efficient. Some of the benefits reported by the participants were: "SKA provided an interactive way of constructing red-black trees and the feedback was good", "SKA helped me in the visualization of a red-black tree" and "SKA is fun and interesting." Furthermore, the experimental results showed that 50% of the respondents requested an undo button while other recommendations for redesign by the respondents were a more attractive user interface, the inclusion of sound in the visualization, and an option to turn off pop-ups.

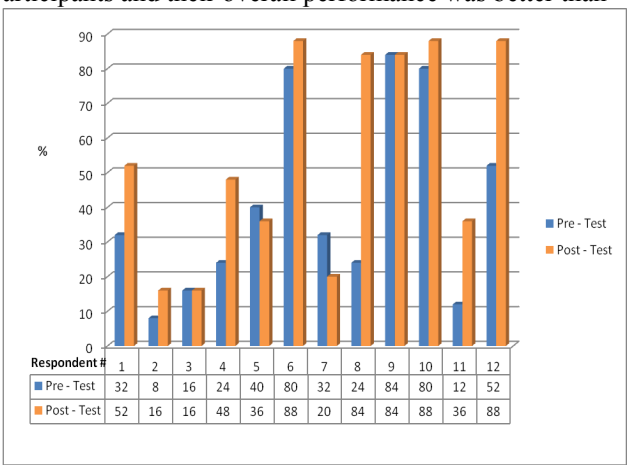
The pre- and post-test results are depicted in Figures 6 and 7. Both groups increased their scores significantly Group A from an average of 40% on the pre-test to 55% on the post-test ($N=12$, $p<0.028$, two-tailed), and Group B from an average of 34% on the pre-test to 69% on the post-test ($N=12$, $p<0.00018$, two-tailed). As a result, this suggests that the AV software along with notes and video-taped lectures can be used to support the learning of RBTs.

E. Discussion

A disparity is shown in the pre- and post-test scores of both groups. This variation might be attributed to several factors such as age group, type of learner (aural or visual), focused attention (not easily distracted), willingness to use the additional resources provided (help file, rule-list, and examples) and the classification of learner (passive or active).

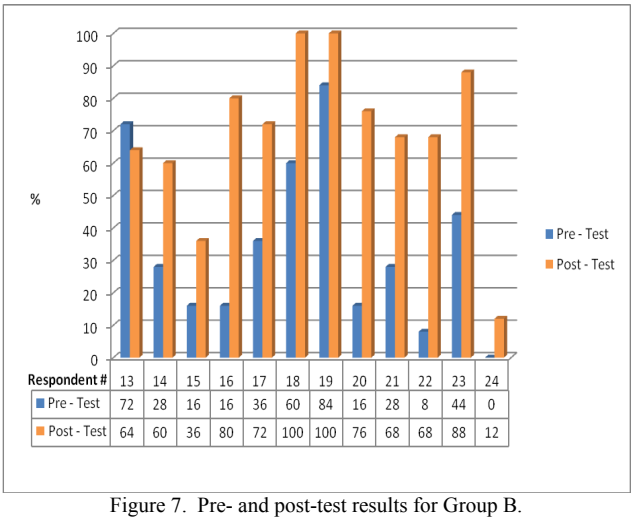
It may be the case that Group B participants were more visually stimulated and focused as the average post-test score increased from an average of 34% to an average of 69% when compared to group A participants (40% to 55%).

The demographics reported in the questionnaire showed that group B participants were much younger than group A participants and their overall performance was better than



Respondent #	1	2	3	4	5	6	7	8	9	10	11	12
Pre - Test	32	8	16	24	40	80	32	24	84	80	12	52
Post - Test	52	16	16	48	36	88	20	84	84	88	36	88

Figure 6. Pre- and post- test results for Group A.



Respondent #	13	14	15	16	17	18	19	20	21	22	23	24
Pre - Test	72	28	16	16	36	60	84	16	28	8	44	0
Post - Test	64	60	36	80	72	100	100	76	68	68	88	12

Figure 7. Pre- and post-test results for Group B.

group A. This may be due to younger individuals having more exposure to technology as observations revealed that they were more receptive to learning and manipulating the software.

Furthermore, active use of the additional resources appeared to have pedagogical value in facilitating the learning of the RBT data structure. For example, observations show that the majority of group B subjects periodically paused the video presentation to ask the researcher questions about RBTs, as compared to group A, where very few students asked questions in the lecture session. Nearly all of group B participants were more receptive to manual construction when compared to most of group A participants who were more easily frustrated and adamantly requested an undo function and automatic rotation of the nodes.

Additionally, it may have been that participants who barely increased or did not improve their scores did not fully understand the basics of the RBT data structure, which suggests that they might have struggled to understand the material presented in the lecture session, videotape, or in the additional resources. Therefore, one would have to conduct a special investigation to address the learning needs of these students.

V. Future Work

Although the SKA extension for RBTs was of high pedagogical value, its current user interface is in need of improvement. As a result, the aesthetic features which support the SKA extension for RBTs will be enhanced in future work.

Currently, the SKA extension for RBTs does not have the capacity to undo or redo operations. This has resulted in frustration and tedious construction of RBTs when using SKA. Therefore, undo and redo functions will be incorporated into the internal structure of SKA.

At this moment, the SKA extension for RBTs does not facilitate the inclusion of sound and video. Consequently, there are future plans to integrate audio and video recordings into the SKA, in order for users to have a more available reference to instruction material. This might be beneficial to distance education domains and intelligent tutoring systems.

In the future, the SKA extension for RBTs will be improved to accommodate variations for different types of learners. This will allow an advanced learner to turn off pop-ups with explanations which were described as "useful but annoying" to these learners. In addition, evaluative questions will range from simple to more complex RBTs and a user profile will be kept which will keep a log of the user performance during evaluation of each student while using the software. As a result, both instructors and students alike can track the performance progress of the students and hence, additional material can be provided based on the needs of the student.

VI. Conclusion

The application of multimedia technology in DS&A courses is not intended to replace the instructor as the focus for teaching the data structure, but to serve as a teaching tool to aid in instruction. In this research, the use of multimedia in computer science education appears to be an effective strategy for learning and reinforcing the rules of RBTs. Also, the application of interactive AVs may foster the development of self-directed learning by engaging students in problem solving while offering feedback and advice. We believe that this strategy can be employed in other domains, and we intend to do so in the future.

Acknowledgment

The authors would like to thank Mr. Yuri Lee, Mr. Carl Beckford and Ms Carol Linton for their assistance with this journal.

References

[1] A. Hamilton-Taylor and K. Davis, "Interactive learning for computer science education via ska algorithm visualisation," in Active and Interactive Teaching and Learning Guide, M.E. (EDs). Kingston, Jamaica: Jamaica Fulbright-Humphrey Alumni Association (JFHAA) in association with the Public Affairs Section of the American Embassy., 2010, pp. 301–326.

[2] C. A. Shaffer, A practical introduction to data structures and algorithms analysis, Java edition. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1998.

[3] R. Baecker, "Sorting out Sorting: A Case Study of Software Visualization for Teaching Computer Science," in Software Visualization: Programming as a Multimedia Experience. Massachusetts, MA, USA: MIT Press, 1998, pp. 369–381.

[4] M. Brown and R. Sedgewick, "A System for Algorithm Animation," ACM SIGGRAPH Computer Graphics, vol. 18, no. 3, pp. 177–186., July 1984.

[5] J. Stasko, "Tango: A Framework and System for Algorithm Animation," IEEE Computer, vol. 23, no. 9, pp. 27–39., Sept. 1990.

[6] B. Price, I. Small, and R. Baecker, "A principled taxonomy of software visualization," Journal of Visual Languages and Computing, vol. 4, no. 3, pp. 211–266, 1992.

[7] M. Brown, "A Taxonomy of Algorithm Animation Displays," in Software Visualization: Programming as a Multimedia Experience. Massachusetts, MA, USA: MIT Press, 1998, pp. 35–44.

[8] C. Hundhausen, "Toward Effective Algorithm Visualization Artifacts: Designing for Participation and Communication in an Undergraduate Algorithms Course," PhD in Computer Science, The University of Oregon, Department of Computer and Information Science, Eugene, Oregon, 1999.

[9] C. Hundhausen, S. Douglas, and J. Stasko, "A Meta-Study of Algorithm Visualization," Journal of Visual Languages and Computing, vol. 13, no. 3, pp. 259–290., June 2002.

[10] J. Urquiza-Fuentes and J. A. Velàzquez-Iturbide, "A survey of successful evaluations of program visualization and algorithm animation systems," Trans. Comput. Educ., vol. 9, no. 2, pp. 1–21, 2009.

[11] D. Jarc, "Assessing the benefits of interactivity and the influence of learning styles on the effectiveness of algorithm animation using web-based data structures courseware," PHD Dissertation, George Washington University, Washington, DC, 1999.

[12] T. L. Naps, G. Rößling, V. Almstrum, W. Dann, R. Fleischer, C. Hundhausen, A. Korhonen, L. Malmi, M. McNally, S. Rodger, and J. Velàzquez-Iturbide, "Exploring the role of visualization and engagement in computer science education," in ITiCSE-WGR '02: Working group reports from ITiCSE on Innovation and technology in computer science education. New York, NY, USA: ACM, 2002, pp. 131–152.

[13] S. Khuri, "Designing Effective Algorithm Visualization" in Program Visualization Workshop, Porvoo, Finland, pp. 1-2, 2001.

[14] E. Kraemer, "Visualizing Concurrent Programs" in Software Visualization: Programming as a Multimedia Experience. Massachusetts, MA, USA: MIT Press, 1998, pp. 237-246.

[15] M.D. Byrne, R. Catrambone and J.T. Stasko, "Evaluating Animations as Student Aids in Learning Computer Algorithms" in Computers & Educ., vol. 33, no. 4, pp. 253-278, 1999.

[16] C. Yeh, "A Framework Proposal for Algorithm Animation Systems," Masters Dissertation, Nelson Mandela Metropolitan University, Republic of South Africa, 2006.

[17] M. Fienup and D. Tesfa, "Algorithm Animation Revisited" in Citeseer, pp. 1-10, 2008.

[18] G. Rößling, M. Schüer, and B. Freisleben, "The animal algorithm animation tool," SIGCSE Bull., vol. 32, no. 3, pp. 37–40, 2000.

[19] T.H. Cormen, C.E. Leiserson and R.L Rivest. "Introduction to Algorithms" Massachusetts, MA, USA: MIT Press, 1990.

[20] S. Papert, Mindstorms: Children, Computers, and Powerful Ideas. New York, NY, USA: Basic Books, Inc., 1980.

[21] A. G. Hamilton-Taylor, "The Study and Design of Algorithm Animations," PhD in Computer Science, The University of Georgia, Computer Science Department, Athens, Georgia, 2006.

[22] H. Beyer and K. Holtzblatt, Contextual Design Defining Customer-Centered Systems. California, CA, USA: Morgan, Kaufmann Publishers Inc., 1998.