Int. J. Human-Computer Studies (2001) **54,** 265–284
doi:10.1006/ijhc.2000.0409
Available online at http://www.idealibrary.com on IDEAL®

Rethinking the evaluation of algorithm animations as learning aids: an observational study

Colleen Kehoe, John Stasko and Ashley Taylor
Graphics, Visualization, and Usability Center College of Computing,
Georgia Institute of Technology, Atlanta, GA 30332-0280, USA.
email: colleen,stasko,ataylor@cc.gatech.edu

One important aspect of creating computer programs is having a sound understanding of the underlying algorithms used by programs. Learning about algorithms, just like learning to program, is difficult, however. A number of prior studies have found that using animation to help teach algorithms had less beneficial effects on learning than hoped. Those results surprise many computer science instructors whose intuition leads them to believe that algorithm animations should assist instruction. This article reports on a study in which animation is utilized in more of a "homework" learning scenario rather than a "final exam" scenario. Our focus is on understanding *how* learners will utilize animation and other instructional materials in trying to understand a new algorithm, and on gaining insight into how animations can fit into successful learning strategies. The study indicates that students use sophisticated combinations of instructional materials in learning scenarios. In particular, the presence of algorithm animations seems to make a complicated algorithm more accessible and less intimidating, thus leading to enhanced student interaction with the materials and facilitating learning.

1. Introduction

One fundamental aspect of being a good programmer is being knowledgeable of the underlying algorithm(s) used in a program. Often, a computer program is nothing more than a concrete, programming language-based implementation of an algorithm. By comprehending a program's underlying algorithm(s), a programmer is better able to understand its data structures, fundamental operations and inherent methodologies. Clearly, this is not the only aspect of being a good programmer, but we feel that it has been an overlooked component of the skills necessary to become a successful programmer.

Unfortunately, there is something difficult about understanding and analysing algorithms; ask any computer science student. What that "something" is and how to reduce the "difficulty" are two problems whose solutions are anxiously awaited by many students and instructors. Meanwhile, guided mainly by intuition, instructors have been looking toward *algorithm animation* (Brown, 1988) as a tool to help their students learn. It is certainly possible to learn about an algorithm without using an animation, but to many it seems almost obvious that students could learn faster and more thoroughly with one: the dynamic, symbolic images in an algorithm animation help provide a concrete appearance to the abstract notions of algorithm methodologies, thus making them more explicit and clear. In addition, students using animations report that they feel the animations assist them in understanding an algorithm (Stasko, Badre & Lewis, 1993). Imagine the surprise of students and instructors when empirical research about the benefits of algorithm animation began to show disappointing results (Stasko *et al.*, 1993; Byrne, Catrambone & Stasko, 1999).

This article is a further step in our examination of the effects of animation on learning about computer algorithms and programs. We are motivated by the disappointing, mixed results of prior studies and a general curiosity about both how and why animation and multimedia technologies can assist instruction. Much prior research in this area has focused on algorithm animation technologies. This work continues our efforts on analytical, cognitive aspects of the domain. Mayer provides a fitting motivational prolog to our efforts:

> "At this time, the technology for multimedia education is developing at a faster pace that a corresponding science of how people learn in multimedia environments. Technological advances in computer-based graphics—including animation—and text-based graphics—including the use of animations—have not been matched by corresponding scientific advances in understanding how people learn from pictures and words." (Mayer, 1997, p. 4).

To begin, we shall briefly review a few of the prior empirical studies of algorithm animation that have provided mixed results. A study conducted by Stasko, Badre and Lewis in 1993 used an interactive animation to teach a complicated algorithm to computer science graduate students (Stasko *et al.*, 1993). Their results showed a "non-significant trend favoring the animation group" in scores on a post-test used to evaluate understanding. The study hypothesized that the animation would aid procedural understanding, but the animation group did not perform any better than the control group on questions testing procedural knowledge. The authors attribute the lack of a performance advantage with animation to a property of most visualizations–that they represent an expert's understanding of the algorithm, not a novice's. "For a student to benefit from the animation, the student must understand both [the] mapping [from the algorithm to the graphics] and the underlying algorithm on which it is based... Students just learning about an algorithm do not have a foundation of understanding upon which to construct the visualization mapping."

A more recent study conducted by Byrne, Catrambone and Stasko (BCS) also found limited effects for undergraduates using interactive animations (Byrne *et al.*, 1999). The study examined the relation of animation to evoking predictions in students. In learning new algorithms, some students viewed animations and some were prompted to make predictions about an algorithm's operation on novel data sets. For a simple algorithm, the use of animation and/or prediction was beneficial on challenging questions, as measured on a post-test. For a more complex algorithm, however, animation and/or prediction provided no significant benefit.

Not all algorithm animation studies have had disappointing results, however. Lawrence's dissertation research uncovered a variety of results, but one particular experiment showed a positive benefit to the use of animations in after-class laboratory sessions when students were allowed to interact with animations by entering their own data sets as input to algorithms (Lawrence, Badre & Stasko, 1994).

Hansen *et al.* built a hypermedia environment with animation as a key component to help teach students about algorithms. The use of the system exhibited significant learning benefits, though these benefits may have been caused by any aspect of the environment, not just animation (Hansen, Schrimpsher & Narayanan, 1998).

The use of animation to help learners has been studied in a broader context than just learning about algorithms. A number of studies have focused on the contribution of animation as an aid to learning in other domains such as physics and user interfaces on computers.

Rieber, Boyce and Assad conducted a study in 1990 using a computer-based science lesson to teach introductory Newtonian mechanics to adults (Rieber, Boyce & Assad, 1990). In short, their results showed that neither the addition of static graphics, nor animated graphics had any effect on learning as measured by a multiple-choice post-test. The study mainly attributes this to a maturation effect: "older students consistently rely less on external images than younger students. (Pressley, 1977)" The claim is that adults can and will generate *internal* images given suitable explanations (which the material provided) and therefore the *external* images, the static and animated graphics, were not necessary for learning. On a more promising note, students who viewed animations were able to complete the post-test in significantly less time than the other students. According to the study, the retrieval process requires students to construct images in short-term memory. They hypothesize that the animations aided students in the retrieval process, "presumably by facilitating the initial encoding".

A study by Palmiter and Elkerton in 1991 compared the use of animated demonstrations, written text, and a narrated animation for teaching users how to operate a particular graphical interface (Palmiter & Elkerton, 1991). They expected, based on the results of earlier studies, that the narrated animation users would perform the best. Animation would aid the initial learning and narration would aid retention and transfer. Their results showed, however, that the performance of the animation-only and narrated animation groups was very similar; both had problems with retention and transfer. They found evidence that users in these two groups may have been simply mimicking the procedures and only processing them superficially. As to why the narrations did not have the effect seen with the written text, they give two possible explanations: that auditory text is processed differently from written text, or that users were not paying attention to the narration well enough to process it thoroughly.

Pane, Corbett and John conducted a study of students learning about time-varying biological processes using multimedia software (Pane, Corbett & John, 1996). They compared a multimedia system that included text, graphics, animations and simulations to a control environment that used only text and carefully selected images. They found little evidence of benefits from the multimedia system, and argue that these kinds of instructional materials must be prepared very carefully. They state, "Merely using animation and simulation capabilities of modern computers does not guarantee improvement in students' learning. Well-designed static graphics and text may be just as effective, and much cheaper to produce and use, than animations and simulations."

In a series of experiments between 1989 and 1994, Mayer *et al.* demonstrated that illustrations (both static and animated) can have a dramatic positive effect on learning under certain conditions. Results from the early experiments, using only static illustrations, showed that students who viewed labeled illustrations showed better explanative recall and problem-solving transfer than students who saw only labels or illustrations or neither (Mayer, 1989). Mayer claims that the labeled illustrations played two roles: guiding students' attention and helping them build internal connections (i.e. connections between ideas in the text, as opposed to connections to previous knowledge).

Another set of experiments in 1991 and 1992 with Anderson, considered the use of animations to help students understand scientific explanations (Mayer & Anderson, 1991, 1992). In the experiments, college students with limited mechanical knowledge viewed animations and/or listened to narrations explaining the operation of a bicycle pump and a hydraulic brake. Students who saw an animation and listened to an explanatory narration outperformed those who did not see the animation on a creative, problem-solving test. In a later experiment, this result was tightened to show that the benefit of animation occurred when it was viewed concurrently with hearing an explanation, not when the two occur contiguously (Mayer & Sims, 1994).

Mayer and his collaborators explain these effects by noting how animation contributes to multiple representations of the problem domain. More specifically, they cite the "*integrated dual-code hypothesis*, adapted from Paivio's dual-coding theory (Paivio, 1990; Paivio, 1991), which posits that learners can build both visual and verbal modes of mental representations as well as connections between them". Further, they cite the importance of simultaneity in different multimedia explanations, claiming that a serial presentation animation and narration makes it more difficult for students to build referential connections between the two presentations.

While these experiments with animation in domains other than algorithms certainly help to inform our studies, we are reluctant to make any direct connections between their results and those to be expected for algorithm animations. All these other studies focused on an animation involving a tangible, visual (usually physical) phenomenon such as a pump, a brake or a user interface on a computer. In these cases, learners have a pre-existing visual basis to draw from and leverage in knowledge construction. Algorithm animations, conversely, provide visualizations to computer data structures and operations which do not have any pre-existing visual basis. So, animation is being used not only to explain a dynamic process, but also to depict entities without existing visual representations. In some sense, algorithm animation is a broader, more abstract and complex problem domain than those studied in these other experiments.

2 Motivation

All of the studies mentioned above either explicitly or implicitly (through their design) test a theory of how animations could aid learning. This theory is reflected in the choice of subject matter, the content of the animation, the accompanying materials, the method of presentation, the evaluation of learning and the tasks and participants chosen (Gurka & Citrin, 1996). In the studies that have failed to find significant benefits to using animation, at least three explanations seem plausible.

- That there are no or only limited benefits from animation.
- That there are benefits, but the measurements used in the studies are not sensitive to them.
- That something in the design of the experiment is preventing participants from receiving the benefits or in other words, the theory of *how* animations could help needs to be re-examined.

This study investigates the third possibility by altering the traditional manner in which animation has been used in empirical studies and by making detailed observations of students using algorithm animations in educational settings. Other researchers, such as Hundhausen and Douglas, have theorized that the manner in which animation has been integrated into empirical studies and the learning assessment methods are inadequate for accurately assessing the benefits of animation (Douglas, Hundhausen & McKeown, 1995, 1996; Hundhausen, 1998). They suggest that the answers to research questions such as "How could animations aid learning?" lie in qualitative data gathered from observing students viewing and interacting with the animations in authentic settings. This is in stark contrast to controlled, comparative studies which usually require settings that are not authentic in order to produce clean, quantitative data.

This article describes a study which is a compromise between the quantitative and qualitative approaches, hopefully leveraging the best points of each. First, the study situates algorithm animations in a learning setting in a much more flexible manner than previous studies, thus accommodating different student uses of the animations. Similarly, students access the learning assessment instruments in the study in a more flexible manner. Second, we make detailed observations of each student, characterizing how they use animations and other instructional materials to learn about different aspects of an algorithm. Finally, we still include a traditional examination-style set of questions to assess how well the students understood the algorithm, which is being presented.

The purpose of this study, then, was to gain insight into the manner in which animation might fit into a successful learning strategy. In particular, we wanted to observe students using animations in a more realistic, homework-style learning situation to determine:

- What learning materials would students use when confronting different types of questions about an algorithm.
- If the availability of animations will influence student learning behaviors and motivation.
- How the learning choices made by students influence their performance on assessment instruments measuring understanding of algorithm concepts.
- If the presence of animations would facilitate student learning about the algorithm.

3 Study design

The topic used in the study was the *binomial heap*, a data structure that can be used to implement an abstract data type is called a priority queue. Priority queues manage a set of nodes with associated key values and are used in many computer science algorithms.

The most basic version of a priority queue involves three operations: *insert*, *extract-minimum*, and *union*. *Insert* simply adds a new node to the priority queue and *extract-minimum* removes and returns the node with the smallest key value. The *union* operation is utilized primarily as a sub-procedure and is performed after each of the others to combine trees of the same size.

The binomial heap and its accompanying algorithms are often taught in advanced undergraduate or graduate level computer science courses. Binomial heaps consist of a forest (ordered set) of binomial trees. Binomial trees are unique in that they always have a size which is a power of two. Binomial trees of equal sizes are combined to make larger trees. The data structure is appealing because all three of its fundamental operations run in logarithmic time. For more details on binomial heaps and their operations, consult any comprehensive computer science algorithms text such as Cormen, Lieserson and Rivest (1990). Note that the binomial heap is one of the algorithms studied in BCS Byrne *et al.* (1999), an experiment in which the use of animation did not provide a significant learning benefit.

Twelve students participated in the study, all volunteers and all graduate students in computer science at the Georgia Institute of Technology. They had had little or no exposure to binomial heaps, but all had taken either undergraduate- or graduate-level algorithms courses. Because of these qualifications, we considered these students "expert learners" and assumed that they have developed successful strategies for learning new material on algorithms.

In previous studies on algorithm animation (Stasko *et al.*, 1993; Lawrence *et al.*, 1994; Byrne *et al.*, 1999) students were divided into two groups. Each group was provided with learning materials such as text, figures and a taped mini-lecture to help learn about some new algorithm. In addition, one of the two groups had access to an algorithm animation about the topic algorithm. The students used these materials to learn about the algorithm for a set period of time, then the learning materials were taken away. At that point, a post-test was administered in which the students had a specific amount of time to answer the questions. The methodology of these experiments simulated an exam scenario.

The methodology of this study differed significantly, simulating more of a homework scenario. Again, the students were divided into two groups and provided with learning materials about binomial heaps, with one group having access to algorithm animations. All the students were given the questions at the start of the session, however, rather than using explicit "learning" and "exam" periods. All the learning materials were available during the entire session as well. Also, no maximum time limit was imposed—each student could work with the materials for as long as they wished. Our hope was that this alternative learning scenario might uncover the effects of algorithm animation more than the traditional exam scenario used in previous studies. Furthermore, we felt that this "homework" scenario would be more likely to encourage the kind of exploration activities we hoped to observe. The two groups of students did have comparable backgrounds with respect to SAT score, GPA and experience in algorithm classes.

The learning material on binomial heaps was adapted from a popular algorithms textbook (Cormen *et al.*, 1990) and presented on a page on the World Wide Web. The explanatory material including exposition, analysis, pseudocode and figures was adapted directly from the book. For the group of learners with algorithm animation access, hotlinks in the text to relevant animations tightly integrated the textual material with the animations. When reading about a particular operation, students could simply click on a hotlink to bring up an animation demonstrating the operation. The animations were implemented using the POLKA animation system (Stasko & Kraemer, 1993). Three predefined animation segments were available to students: one illustrated the combination of two small binomial trees into a larger tree and the other two illustrated the *extract-minimum* and *union* operations. Each animation was modeled after a series of static illustrations from the textbook. The animations contained the same basic images as the static illustrations, supplemented by in-between frames providing a smooth transition through the images thereby making the relationship between objects in each image more explicit. Binomial heap operations involve fairly complex movements of nodes and subtrees and the animations smoothly illustrated these steps. The animations allowed participants to step forward or backward through the steps of the operation. They could also be displayed simultaneously with the textual material. Figures 1 and 2 present still frames from the *extract-min* animation. Each of these frames corresponds to one of the key points of the operation. Note, however, that many in-between transitional frames between consecutive pairs of figures here are not shown, for brevity.

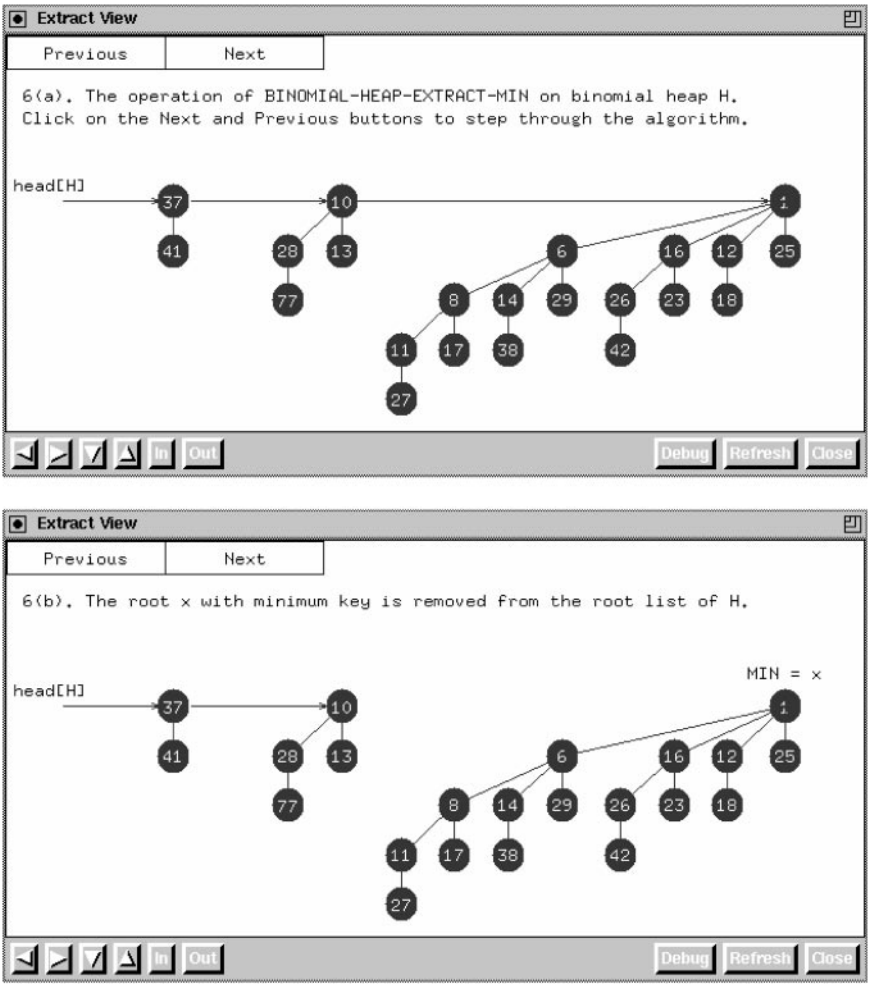
For the "non-animation" group, the learning materials included still figures of the operations' key points, but no algorithm animation. We wanted to equate the two groups' learning materials as closely as possible, while making the only difference be the availability of the algorithm animations.

The questions to be answered by the students covered various aspects of binomial heaps including operations, definitions, mathematical properties and running times. Most questions tested recall (e.g. questions on heap properties, complexity and form) or application (e.g. run an operation on new data). The final question was more of a synthesis style analogy question. Details on the questions can be found later in Section 4. The questions actually were taken from the post-test used in the BCS study (Byrne *et al.* 1991). Students were encouraged to verbalize their thought processes while viewing the materials and working on the questions. We wanted to observe students using the materials to discover what strategies they employed while trying to answer questions about binomial heaps. In all sessions, student activity and computer screen activity were video (and audio) taped. After each session, we also informally discussed what occurred in the session with the student and administered a short questionnaire about the session.

4 Results

We will begin describing the study's results by discussing how the two groups performed quantitatively, that is, with respect to exam score and time taken. Following that, we will include a more qualitative analysis of the sessions.

Figure 3 lists the exam scores of the twelve students in the two groups. The animation group had a higher average, 20.5 vs. 16 correct replies, out of a total of 23 questions. Two students from the animation group had perfect scores and four of the animation group students scored higher than the top non-animation group student. Note that the exam



6(a). The operation of BINOMIAL-HEAP-EXTRACT-MIN on binomial heap H.
Click on the Next and Previous buttons to step through the algorithm.

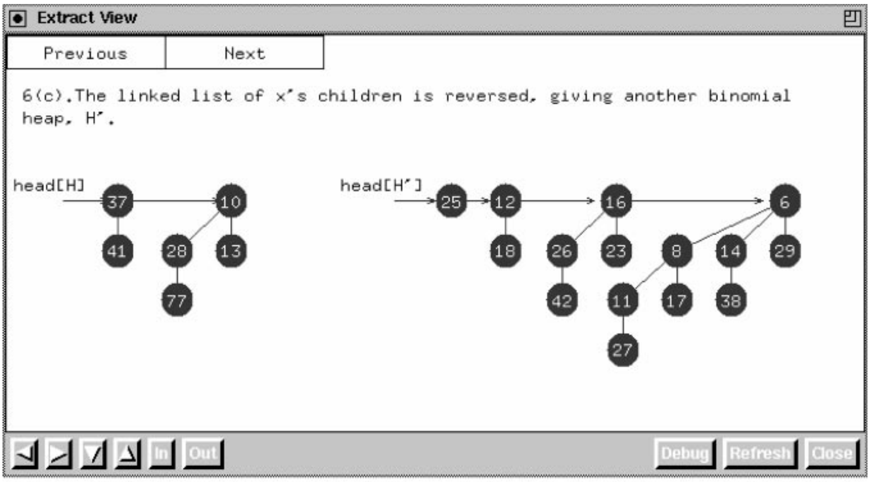
6(b). The root x with minimum key is removed from the root list of H.

FIGURE 1. Extract-min animation frames.

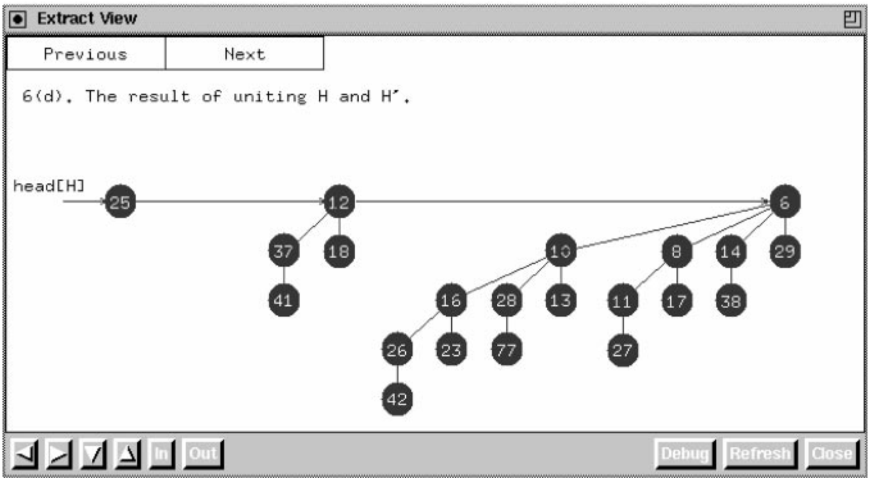
score difference between the two groups is significant, $t(10) = 2.55$, $p < 0.029$ (two-tailed). Our focus here is not to dwell on statistics, however. Rather, we seek to better understand how animations can affect students' learning processes.

Figure 4 reports the results of the exam on a question-by-question basis. The exam questions can be divided into groups of comparable style questions:

1a–f True/False questions about the properties of binomial heap structures and operations.
2–3 Questions about the definition and form of the binomial heap data structure.
4–5 Questions about the computational complexity (running time) of binomial heap operations.



6(c).The linked list of x's children is reversed, giving another binomial heap, H'.



6(d). The result of uniting H and H'.

FIGURE 2. Extract-min animation (continued).

6–9 Questions about the definition and form of the binomial heap data structure (similar to questions 2 and 3).
10–11a,b Analytical questions about abstract, general properties of binomial heap operations.
12–15 Questions requiring students to carry out a specific, example binomial heap operation.
16 Question about the analogy between binomial heaps and binary arithmetic.

The two groups of students performed comparably on the questions except for three particular styles of questions. To a small degree, performance varied on the initial T/F heap property questions. On questions 1a, d and f the non-animation group had two

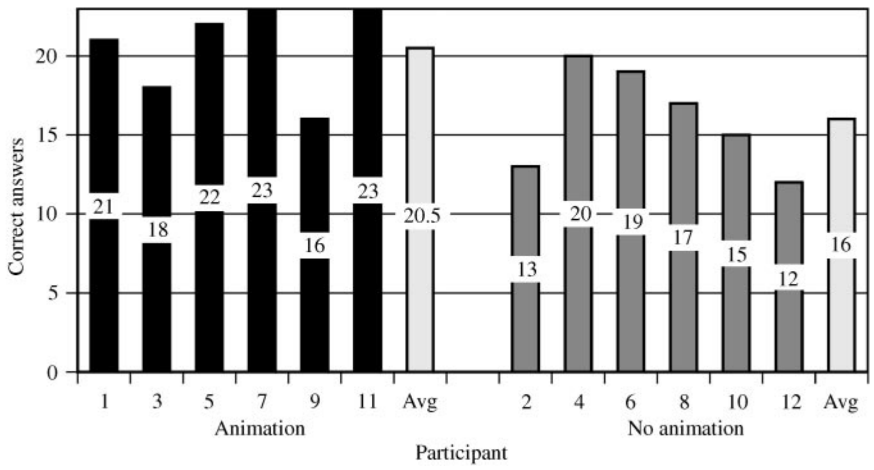


FIGURE 3. Summary scores of students from both groups on the examination.

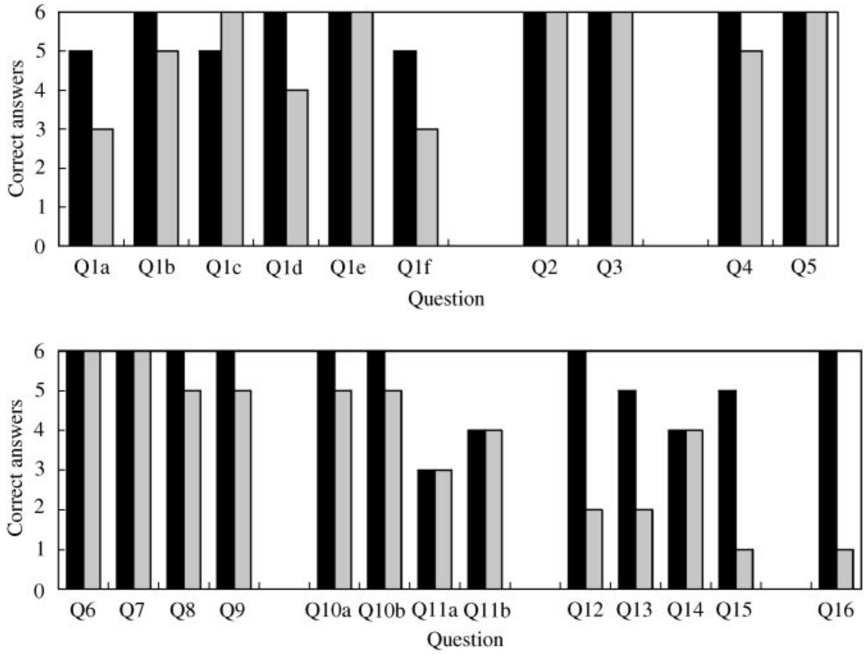


FIGURE 4. Number of correct replies of the two groups per question: ■, animation; □, no animation.

more incorrect answers than the animation group. This difference is probably not important given the small number of students involved. More noticeably, however, the animation group more clearly outperformed the non-animation group on questions 12–15 in which the students had to carry out operations on example binomial heaps.

TABLE 1
Number of different students per condition who referenced a particular learning material while answering questions 12–15

	Animation	Non-animation
Q 12	3 animation	6 static images
	2 pseudocode	1 pseudocode
	1 text	3 text
	0 figures	6 figures
Q 13	3 animation	2 static images
	3 pseudocode	3 pseudocode
	1 text	2 text
	0 figures	3 figures
Q 14	2 animation	3 static images
	2 pseudocode	2 pseudocode
	1 text	2 text
	0 figures	3 figures
Q15	2 animation	0 static images
	1 pseudocode	0 pseudocode
	1 text	0 text
	1 figure	2 figures

Non-animation performance on these questions was poor, missing a majority, while the animation students did quite well. This result is not surprising to us in that it would appear that the animations would most help learners master the basic mechanics of heap operations, that is, how the steps are carried out. Finally, all the animation students correctly answered question 16 about the correspondence between binomial heaps and binary arithmetic. Only one of the non-animation group students answered this question correctly.

In an effort to analyse the performance on the procedural questions 12–15 more closely, we examined the materials referenced by each student as he or she answered these questions. In general, the students in both groups referred back to materials more for the earlier questions, then they tended to answer the later questions without assistance. On questions 12–15 respectively, one, one, two and three of the animation group students answered without referring to any learning materials. Of the non-animation students, zero, two, three and four students did not refer to any learning materials, respectively. For the students who referred back to the learning materials, Table 1 summarizes which materials were used. Recall that a student was able to look at any or all of the media for assistance.

Note how the animation group did make moderate use of the animations and pseudocode, while only once using a figure. The non-animation group accessed the static images moderately, and the figures much more than the animation group did. Both groups accessed the pseudocode and text at roughly similar levels.

Now let us turn our attention to the amount of time used by each participant. Recall that the students could work for as long as they wanted. Figure 5 and Table 2 present the total time taken by each student, the time spent during preparatory learning (before

TABLE 2

ID	Preparatory time	Question time	Review time	Total time	Score
Animation group					
1	23.5	42.5	12.0	78.0	21
3	14.0	44.5	12.5	71.0	18
5	19.5	43.5	5.0	68.0	22
7	39.0	43.0	2.0	84.0	23
9	30.0	41.5	20.5	92.0	16
11	23.5	54.5	34.0	112.0	23
Avg.	24.9 (8.9)	44.9 (4.8)	14.3 (11.6)	84.2 (16.2)	20.5 (2.9)
Non-animation group					
2	1.0	27.0	12.0	40.0	13
4	18.5	59.0	7.5	85.0	20
6	27.5	25.0	8.0	60.5	19
8	55.0	55.5	11.5	122.0	17
10	10.0	45.5	2.5	58.0	15
12	6.0	63.5	11.0	80.5	12
Avg.	19.7 (19.7)	45.9 (16.5)	8.8 (3.6)	74.3 (28.5)	16.0 (3.2)

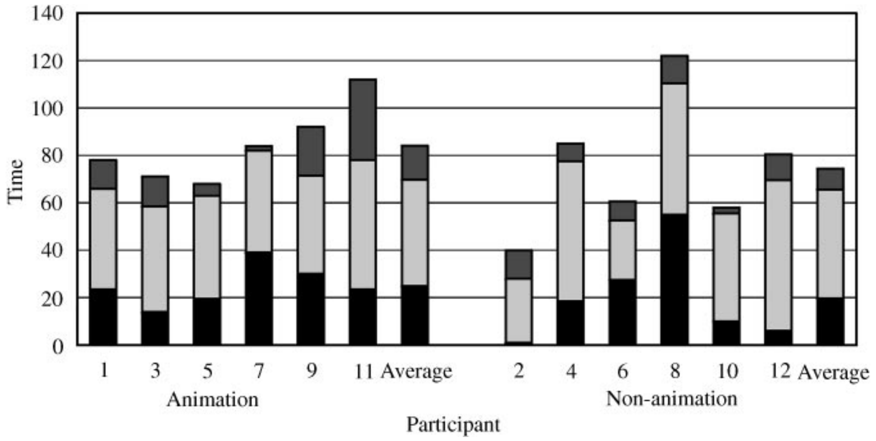


FIGURE 5. Preparatory, question, review and total times for participants. Standard deviations (S.D.) are indicated in parentheses: ■, review time; □, question time; ▩, preparatory time.

attempting to answer any questions), the time spent actually working on questions, and "review" time.† Review time is defined as time spent reviewing the learning materials informally, after a student had started to answer questions.

† These time figures are as accurate as we could determine from logs and videotapes of the sessions. Nonetheless, a small amount of variance should be allowed given the difficulties of precisely assessing start and stop points of different activities.

TABLE 3

Time spent by the students accessing the different learning materials. For the non-animation group, the Anim column indicates use of the static figures taken from frames of the animation as explained earlier. Standard deviations are indicated in parentheses

ID	Text time	Code time	Figure time	Anim time	Joint time	Total time	Score
Animation group							
1	23.5	11.5	5.5	17.0	21.0	78.0	21
3	29.5	8.5	5.5	12.5	28.0	71.0	18
5	22.0	6.0	10.0	19.0	40.5	68.0	22
7	17.0	10.5	18.5	18.0	32.5	84.0	23
9	34.0	18.0	6.0	19.0	37.0	92.0	16
11	46.5	41.5	6.0	19.0	65.0	112.0	23
Aug.	28.8 (10.5)	16.0 (13.1)	8.6 (5.2)	17.4 (2.5)	37.3 (15.2)	84.2 (16.2)	20.5
Non-animation group							
2	6.5	0.5	5.5	15.5	16.0	40.0	13
4	32.0	17.0	11.5	31.5	57.0	85.0	20
6	23.5	13.0	4.0	4.5	11.5	60.5	19
8	62.5	18.0	27.5	29.0	59.0	122.0	17
10	26.5	12.5	2.5	29.0	43.5	58.0	15
12	22.5	2.5	13.5	16.5	44.5	80.5	12
Avg.	28.9 (18.5)	10.6 (7.4)	10.8 (9.3)	21.0 (10.6)	38.6 (20.3)	74.3 (28.5)	16.0

The animation group averaged almost 10 min longer per session than the non-animation group—84.2 to 74.3 min—though this difference is not statistically significant as the times were quite variable. The difference in averages is perhaps even more noteworthy when one considers the particularly long session, 122 min, of student 8 in the non-animation group. Similarly, the animation group had higher average times for both the preparatory and review segments, though again these differences were not statistically significant. The particularly long preparation time of student 8 again makes the difference in that segment seem not as large as it could have been. Interestingly, the averages for question time were nearly identical between the two groups, with the animation group being particularly consistent. So, in summary, we see that the animation group students spent longer in the session on average and this difference largely resulted from time spent studying the learning materials, not answering the questions.

Table 3 lists the time spent examining each of the styles of learning material, both exclusively and in combination with other material(s). Also listed is total "Joint time" meaning time spent using *any* two or more of the materials closely together, often even positioned side by side. A number of items of interest stand out in this data: first, note the consistency of time spent viewing animations by the first group where all but one of the participants viewed animations between 17 and 19 min. Contrast this with a relatively high variability by the non-animation group in the use of the static images (listed under "Anim") and figures. Both groups exhibited fairly high variability in the use of pseudocode. Student 11 in the animation group far exceeded all others here. Joint times among both groups varied moderately. Fundamentally, the times in this table help reflect the variety of learning styles across different students. On the whole, it appears that the animation group exhibited more consistency in their use of learning materials throughout the sessions.

4.1 SESSION EXPERIENCES—GROUP WITH ANIMATION

The students' quantitative performance during the sessions was merely one component of this study. Equally important to us was *how* the sessions went, what the students did during the sessions and the feedback they provided about the learning materials. In this section, we describe observations made about the student sessions and some key incidents from the sessions. To begin with, we discuss the sessions with the 6 students who had algorithm animations present.

Virtually all of these students made extensive use of a variety of different types of learning materials both while initially learning about the binomial heaps and while answering the questions. Five of the six students positioned an animation or animations up next to corresponding text or pseudocode about the heap operations, and examined both carefully.

Student 9 was typical. When he first used the union animation, he viewed it for a minute, then switched to the union pseudocode, looking at the cases. He then placed the animation beside the code and compared them saying, "so case 2 applies because there are multiple things of the same degree, or... [looked at the code for a minute, shook head negatively, then found the union text further down]. Ah, here we go. [Reads for a minute.] So that's why case 2 and case 3 are the same here". He then switched between the animation and text in turns every 10–30 seconds for a few minutes, then read text for another 2 minutes.

Interestingly, all the students except 9 here slowed down the animation speed noticeably. Student 9 was the only one to run it at the default moderately fast speed.

Different students tended to favor different media more while they were answering the questions. Student 11, for example, tended to favor the pseudocode. The post-session discussion with her included the following exchange: (The "I" and "S" below refer to interviewer and student, respectively.)

I: I noticed that when you were doing the questions you used the code. Did you find the code more useful at that point?
S11: The reason I was using the code is because I could not remember how it is supposed to do certain procedures. But both are useful because you have to look at the code first to see how it is supposed to work, and then the animation shows you how it actually does it.

Student 7 stood out in using a different methodology while answering questions. He utilized all learning materials including animation in the pre-question preparatory time and to answer the very first exam question. For all subsequent questions he only occasionally referred to textual definitions and figures, however, and did not look at any other materials. For the most part, he simply answered the questions directly. This student did get all 23 questions correct, so clearly he was able to learn about and remember the features of binomial heaps in the initial preparation time.

In the post-session interview and questionnaire, a number of students commented that they felt the animations were helpful in learning.

> **I**: What did you think...?
> **S1**: [Interrupts] I liked it. I liked it. Because I've had a couple classes on this stuff and most of the time I'm like... "What are you talking about?" But this time I actually got it.
>
> Maybe it's because this time it happened to click, but I think the animations helped. It was good to look...to have the algorithms on one hand and the animations on the other, and to [points finger from left to right before him repeatedly] pick it apart like that. That helped a lot.
>
> When I take 6155 [the graduate algorithms class] hopefully this will be ready for me, and you will have a whole book on these, and I can get an "A" and graduate. [Laughs]

This student also wrote, "The animations definitely helped me. The animations provide an example to draw analogies and ideas from. The animations also provide a step by step progress to compare with the algorithm."

Student 3 commented about the animation being a kind of useful but not essential memory helper: "[the animations] helped remind me that the roots are pulled up when you extract a node, but I probably would have figured it out anyway".

Student 7 also felt that the animations were helpful and commented, "I think the animations were definitely useful...it took me a lot of reading to remember all this, but to be able to match up the static [text] cases here when you pick an example in the animation that showed the different [union] cases. This and the extract min operation were good cases for animation." He added, "seeing a series of static pictures is good because they are all there at once. An animation makes it easier to notice the changes between the steps shown in a series of static figures".

Student 11 echoed this sentiment: "Some algorithms are too difficult to picture in [the] mind. Illustration like this is helpful".

In contrast, student 5 was not quite so emphatical about the animations. She said, "The animations were helpful for some of the more complex algorithms [operations], but I'm not sure if the fact that it was animated helped than just having a diagram... I guess diagrams in general are useful."

The animations shown were specific examples of individual operations using prepared heaps and data. One student, 3, commented that he would have preferred to have a complete, general purpose animation of the algorithm available so that he could try out his own data: "If the animation is programmable, so that you can test different situations, this helps to clarify the algorithm, but you still have to be able to read the pseudocode or definition. So this was only helpful for the cases animated."

4.2. SESSION EXPERIENCES—GROUP WITHOUT ANIMATION

Now, let us turn to the 6 students who had supplementary static images, rather than animations, available during their sessions. Like the students in the animation sessions, these students referred to combinations of media often in both the preparatory and question answering portions of their sessions. Again, different students exhibited unique preferences among the materials. The students did extensively refer to the figures and static images, however.

One key difference we observed in this group concerned the students' manner and behavior during the sessions. On the whole, these students were quieter, more serious and seemed to be concentrating more intently during their sessions. They made comments reflecting the challenge of learning about binomial heaps and their relative struggles.

S2: "Do we need to answer all of these [questions]? Some of these I have no idea about."
S8: "I just can't read once and absorb all of what I need to know because there's all these different cases here."
S10: "I think I didn't do too well."
S12: (Humorously) "This is like torture."

Not all the students acted this way, however. In contrast, student 4 was quite relaxed throughout the entire session, even whistling frequently. The stark contrast with the others made his different mood even more striking.

After the sessions concluded, all of these students were shown the animations that the other group had used. We observed how they viewed the animations and elicited their reactions. Student 2 commented, "I think this definitely would have helped my cause." And after seeing another animation he said, "I just could not figure out how to do this heap-union thing. I tried a couple of times. I could have gotten it from the diagrams, but there are just too many nodes."

After viewing the union animation, student 4 noted, "See, this would have taken away all the confusion about why does it progress down [points at heap]. It wasn't until I worked out the next problem that I realized that if you did it, it would have violated...you have a B2, then a B1 tree." When asked further about this reaction to the animations, he commented, "In general, if there is ever an example or animation, I will definitely try to understand by using the example. Because to understand the code and do the iterations is crazy." He then gives low-level details of why an animation helped to explain how a particular operation worked. He finished these comments with, "I think the animations definitely help in understanding it. It could just be the fact that I got it so I can see much more quickly with the animations, but I think that if I had it while I was doing it, especially in the situation [above], it definitely would have helped."

Student 12 echoed the earlier comments of animation session student 3 in wanting a general, complete animation: "It's [the animation] pretty cool. The only thing I would like is to put in a tree that I'm interested in and see what happens."

Another student, 10, detailed how he felt animation may or may not help: "I was trying to figure the steps between figures by drawing them—the animation would have helped because it shows the steps...I can understand more the steps to understand the algorithm, but I don't think it can show you the best and worst case running time. But, given a heap and an operation on the heap, I think it is very useful. That's what I was trying to do. I was trying to figure out how it [the series of static figures] was modified. The animation is more like what you are trying to think...Perhaps [useful to] prove or state mathematical properties, but do the whole analysis of algorithms?—No. To understand the algorithm, how it works?—Yes. Instead of doing it on paper, the computer shows you how it works." He wrote on the questionnaire, "when one is trying to figure out how it works, one forms an animation in one's brain. The animation tries to show the steps as the data structure is updated."

Student 6 had the strongest negative opinion about animation of this group. He wrote, "animation by itself is not very useful. With the support of the web page, it is good, but I think if users work out the examples, it helps a lot. Also, it will be hard to realize the complexity [meaning running time] of the algorithm through animations."

5 Conclusion

This article describes a study of university students learning about a computer algorithm and data structure both with and without the aid of algorithm animations. It involved a more open homework-style learning scenario rather than a closed, exam-style scenario. Students simply were given learning materials and a set of questions, and then told to work as long as they wanted. Student participants in the study were high-ability computer science graduate students, and our findings should be interpreted therein. It is not clear if these results would be replicated on a less experienced, more general student population.

The goal of this study was to gain an understanding of how to use algorithm animations in learning situations and to then inform subsequent experiments exploring the pedagogical value of animations. We feel that the study has met these goals. In particular, it has helped us to formulate three key hypotheses about algorithm animations. Although posed as hypotheses here, we do believe these conjectures to be true. Observations from the study support the hypotheses, as we will argue below. Hopefully, these observations and conjectures will be used to guide further empirical studies in this domain.

> Hypothesis 1: The pedagogical value of algorithm animations will be more apparent in open, interactive learning situations (such as a homework exercise) than in closed exam-style situations.

The majority of prior empirical studies on algorithm animation mimicked exam scenarios: animations were available during training, but then were removed during a post-test. Also, students were given a preset maximum amount of time to work on the post-test. This study, conversely, made the animations available while students were answering questions and allowed unlimited time. From our observations, the students were better able to take advantage of the animations in this more "homework" style learning scenario.

By receiving the questions up front, the students understood the learning goals and objectives better than being just given a large corpus of material and told to "Learn this." Consequently, the students were at a point where they could take advantage of the strengths of the different media better. And it was clear that the animations were used in subtle but important combinations with other learning materials.

We speculate that algorithm animations are not so useful pedagogically when used in isolation. They require careful coordination with other learning materials, or better yet, accompanying (human) instruction that explains how an animation simulates an algorithm's operations.

Hypothesis 2: Even if animations do not contribute to the fundamental understanding of an algorithm,† they do enhance pedagogy by making an algorithm more accessible and less intimidating, thus enhancing motivation. In that regard, they facilitate learning.

We feel that the participants in this study who had algorithm animations available did learn about binomial heaps better than those participants without animations available. The exam scores tend to support this view, but our belief is more founded on observations of the students and our subsequent interactions with them. The animation group was simply more relaxed, more confident in their knowledge and more open to learning.

Students in the animation group spent a longer amount of time in the sessions on the average and this time difference occurred from studying the learning materials, not from answering the questions. The animation students simply seemed more comfortable with the binomial heap materials. In contrast, the non-animation group of students in general seemed more stressed by the learning challenge. They labored more and made more comments about how difficult the algorithm was. They were more likely to stop the session earlier. In fact, a number of these students appeared to make half-hearted attempts at some of the last few questions on the exam.

We feel that algorithm animations can make algorithms less intimidating, and hence more accessible to students, thus enhancing motivation. The animations engage students, making learning be more of an interactive experience than a challenging chore. Consequently, the use of algorithm animations will lead to increased time on task, thus facilitating learning, particularly so with complex, challenging subject matter.

Hypothesis 3: Algorithm animation can best facilitate learning of the procedural operations of algorithms.

In this study, students from both the animation and non-animation groups performed similarly on most of the exam questions. One notable difference occurred on questions about concrete instances of the insert, union and extract-min operations on specific examples of heaps. On those questions (primarily numbers 12–15), the animation students clearly outperformed the non-animation students. Algorithm animations seem best suited to helping to convey the procedural step-by-step operations of an algorithm. They provide an explicit visual representation of an otherwise abstract process.

Note that this is just one attribute of "learning about an algorithm." In addition to procedural methodology, instructors want students to understand the computational complexity of an algorithm, its high-level methodology, how to program it, how it relates to other algorithms and so on. Can algorithm animations facilitate these other learning objectives? Furthermore, what about the retention issue? Do algorithm animations aid retention of important concepts and methodologies? These are challenging questions and the answers are not clear. However, we do feel that future empirical studies examining the potential learning benefit in algorithm animations first focus on the animations' ability to facilitate understanding of algorithms' procedures and operations.

Clearly, the use of computer multimedia technologies such as animation is growing throughout the educational community. Unfortunately, our understanding of how these

† We do not necessarily believe this. See Hypothesis 3.

technologies can be best used is lacking. Mayer well captures our current state in the following quote:

> "The potential for computer-based aids to learning environments remains high, although the current contribution of technology to pedagogic innovation is frustratingly low. Instructional development is too often based on what computers can do rather than on a research-based theory of of how students learn with the technology. In particular, the visual-based power of computer technology represents a grossly underutilized source of potential educational innovation." (Mayer, 1997, p. 17).

We feel that this study provides a valuable step in understanding how animations can assist students in learning within the challenging discipline of computer algorithms. Subsequent studies and experiments should leverage the knowledge gained here and confront the open questions remaining in this domain.

References

Byrne, M. D., Catrambone, R. & Stasko, J. T. (1999). Evaluating animations as student aids in learning computer algorithms. *Computers & Education*, **33,** 253–278.

Brown, M. H. (1988). Perspectives on algorithm animation. *Proceedings of the ACM SIGCHI '88 Conference on Human Factors in Computing Systems*, pp. 33–38. Washington, DC, May.

Cormen, T. H., Leiserson, C. E. & Rivest, R. L. (1990). *Introduction to Algorithms.* Cambridge, MA: MIT Press.

Clark, J. M. & Paivio, A. (1991) Dual coding theory and education. *Educational Psychology Review*, **3,** 149–210.

Douglas, S., Hundhausen, C. & McKeown, D. (1995). Toward empirically-based software visualization languages. *Proceedings of the IEEE Symposium on Visual Languages*, pp. 342–349. Darmstadt, Germany, September.

Douglas, S., Hundhausen, C. & McKeown, D. (1996). Exploring human visualization of algorithms. *Proceedings of Graphics Interface '96*, pp. 9–16. Toronto, May.

Gurka, S. & Citrin, W. (1996). Testing effectiveness of algorithm animation. In *Proceedings of the IEEE Symposium on Visual Languages*, pp. 182–189. Boulder, CO, September.

Hansen, S., Schrimpsher, D. & Hari Narayanan, N. (1998). Learning algorithms by visualization: a novel approach using animation-embedded hypermedia. *Proceedings of the International Conference of the Learning Sciences*. pp. 125–130. Atlanta, GA, December.

Hundhausen, C. D. (1998). *Toward effective algorithm visualization artifacts*. Unpublished Ph.D. Thesis, Department of Computer and Information Science, University of Oregon.

Lawrence, A. W., Badre, A. M. & Stasko, J. T. (1994). Empirically evaluating the use of animations to teach algorithms. *Proceedings of the IEEE Symposium on Visual Languages*, pp. 48–54. St. Louis, October.

Mayer, R. E. & Anderson, R. B. (1991). Animations need narrations: an experimental test of a dual-coding hypothesis. *Journal of Educational Psychology*, **83,** 484–490.

Mayer, R. E. & Anderson, R. B. (1992). The instructive animation: helping students build connections between words and pictures in multimedia learning. *Journal of Educational Psychology*, **84,** 444–452.

Mayer, R. E. (1989). Systematic thinking fostered by illustrations in scientific text. *Journal of Educational Psychology*, **81,** 240–246.

Mayer, R. E. (1997). Multimedia learning: are we asking the right questions? *Educational Psychologist*, **32,** 1–9.

Mayer, R. E. & Sims, V. K. (1994). For whom is a picture worth a thousand words? Extensions of a dual-coding theory of multimedia learning. *Journal of Educational Psychology*, **86,** 389–401.

Paivio, A. (1990). *Mental Representations*: *A Dual Coding Approach.* New York: Oxford University Press.

Pane, J. F., Corbett, A. T. & John, B. E. (1996). Assessing dynamics in computer-based instruction. *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, pp. 197–204. Vancouver, BC, Canada, April.

Palmiter, S. & Elkerton, J. (1991). An evaluation of animated demonstrations for learning computer-based tasks. *Proceedings of the ACM SIGCHI '91 Conference on Human Factors in Computing Systems*, pp. 257–263. New Orleans, LA, May.

Pressley, M. (1977). Imagery and children's learning: putting the picture in developmental perspective. *Review of Educational Research*, **47,** 585–622.

Rieber, L. P., Boyce, M. J. & Assad, C. (1990). The effects of computer animation on adult learning and retrieval tasks. *Journal of Computer-Based Instruction*. **17,** 46–52.

Stasko, J., Badre, A. & Lewis, C. (1993). Do algorithm animations assist learning? An empirical study and analysis. *Proceedings of the INTERCHI '93 Conference on Human Factors in Computing Systems*, pp. 61–66. Amsterdam, Netherlands, April.

Stasko, J. T. & Kraemer, E. (1993). A methodology for building application-specific visualizations of parallel programs. *Journal of Parallel and Distributed Computing*, **18,** 258–264.